DSA 8070 R Session 6: Reduced-Rank Regression and Repeated Measures Analysis

Whitney Huang, Clemson University

Contents

Reduced-Rank Regression

Multivariate Regression Refresher

We model a multivariate response matrix $Y \in \mathbb{R}^{n\times q}$ with predictors $X \in \mathbb{R}^{n\times p}$:

$$Y = XB + E,$$

where $B \in \mathbb{R}^{p\times q}$ is the coefficient matrix.

A naive approach fits each response column separately (OLS for each). But this **ignores cross-response correlation**.

```r
set.seed(123)
n <- 200; p <- 6; q <- 4

# Design matrix with intercept
X <- cbind(1, matrix(rnorm(n * (p - 1)), n, p - 1))
colnames(X) <- c("Intercept", paste0("X", 1:(p - 1)))

# True low-rank coefficient: rank = 2
r_true <- 2
```

```r
A <- matrix(rnorm(p * r_true), p, r_true)
C <- matrix(rnorm(r_true * q), r_true, q)
B_true <- A %*% C  # p x q

# Correlated errors across responses
Sigma_eps <- matrix(c(1.0, 0.6, 0.3, 0.2,
                      0.6, 1.0, 0.5, 0.4,
                      0.3, 0.5, 1.0, 0.7,
                      0.2, 0.4, 0.7, 1.0), q, q)
library(MASS)
E <- MASS::mvrnorm(n, mu = rep(0, q), Sigma = 2 * Sigma_eps)

Y <- X %*% B_true + E
colnames(Y) <- paste0("Y", 1:q)

# Fit separate OLS for each response (baseline)
ols_fits <- vector("list", length = q)
for(j in 1:q){
  ols_fits[[j]] <- lm(Y[, j] ~ X - 1)  # lm adds intercept
}
names(ols_fits) <- colnames(Y)
#Compare the estimate and the true parameter
B_est <- array(unlist(lapply(ols_fits, coef)), dim = c(p, q))

B_est
```

```
##            [,1]      [,2]       [,3]       [,4]
## [1,] 1.29633691 0.3063739 -0.5678854  0.1460612
## [2,] 2.32378304 0.6533733 -3.0533527  4.0340122
## [3,] 0.36436374 0.1528000 -0.8591614  1.1869485
## [4,] 0.07241715 0.1164822  0.6869218 -0.7811949
## [5,] 4.43669300 1.1061847 -4.1532537  4.1316973
## [6,] 0.03901217 0.4363386 -3.4027784  5.2680089
```

```r
B_true
```

```
##             [,1]        [,2]       [,3]       [,4]
## [1,]  1.33072102  0.25707090 -0.5417158  0.1524423
## [2,]  2.40068397  0.67164166 -3.2856474  3.9295441
## [3,]  0.34329260  0.13196539 -0.8687432  1.1934462
## [4,] -0.04855480 -0.05568266  0.5339427 -0.8195902
## [5,]  4.41898000  1.06179603 -4.1101136  4.1652634
## [6,]  0.06024535  0.30995319 -3.3372124  5.2514408
```

Reduced-Rank Regression (RRR): SVD-Based Estimator

Idea: Fit full multivariate OLS, then project fitted values to rank r via SVD.

Algorithm:

1. $\widehat{B}^{OLS} = (X^\top X)^{-1} X^\top Y$
2. $\widehat{Y} = X\widehat{B}^{OLS}$

3. SVD: $\widehat{Y} = U\Sigma V^\top$
4. Keep top r right singular vectors V_r, set $\widehat{B}^{RRR} = \widehat{B}^{OLS} V_r V_r^\top$

```r
RRR_fit <- function(X, Y, r){
  # X: n x p, Y: n x q
  XtX <- crossprod(X)
  ridge <- 1e-8 * diag(ncol(X))
  B_ols <- solve(XtX + ridge, crossprod(X, Y))  # p x q
  Yhat  <- X %*% B_ols                           # n x q
  svdY  <- svd(Yhat)
  Vr    <- svdY$v[, 1:r, drop = FALSE]           # q x r
  B_rrr <- B_ols %*% Vr %*% t(Vr)                # p x q
  out <- list(B_ols = B_ols, B_rrr = B_rrr, Vr = Vr, svd = svdY)
  return(out)
}

predict_multivar <- function(Xnew, B){
  Xnew %*% B
}
```

Try a few ranks and compare in-sample fit:

```r
r_grid <- 0:min(ncol(X), ncol(Y))  # from 0 up to min(p, q)
RSS_by_r <- matrix(NA_real_, nrow = length(r_grid), ncol = 2)
colnames(RSS_by_r) <- c("r", "RSS")

for(i in seq_along(r_grid)){
  r <- r_grid[i]
  if(r == 0){
    B <- matrix(0, nrow = ncol(X), ncol = ncol(Y))
  } else {
    B <- RRR_fit(X, Y, r)$B_rrr
  }
  Yhat <- X %*% B
  RSS_by_r[i, ] <- c(r, sum((Y - Yhat)^2))
}
RSS_by_r
```

```
##      r       RSS
## [1,] 0 26482.308
## [2,] 1  4007.351
## [3,] 2  1495.498
## [4,] 3  1479.386
## [5,] 4  1473.253
```

Rank Selection via Cross-Validation

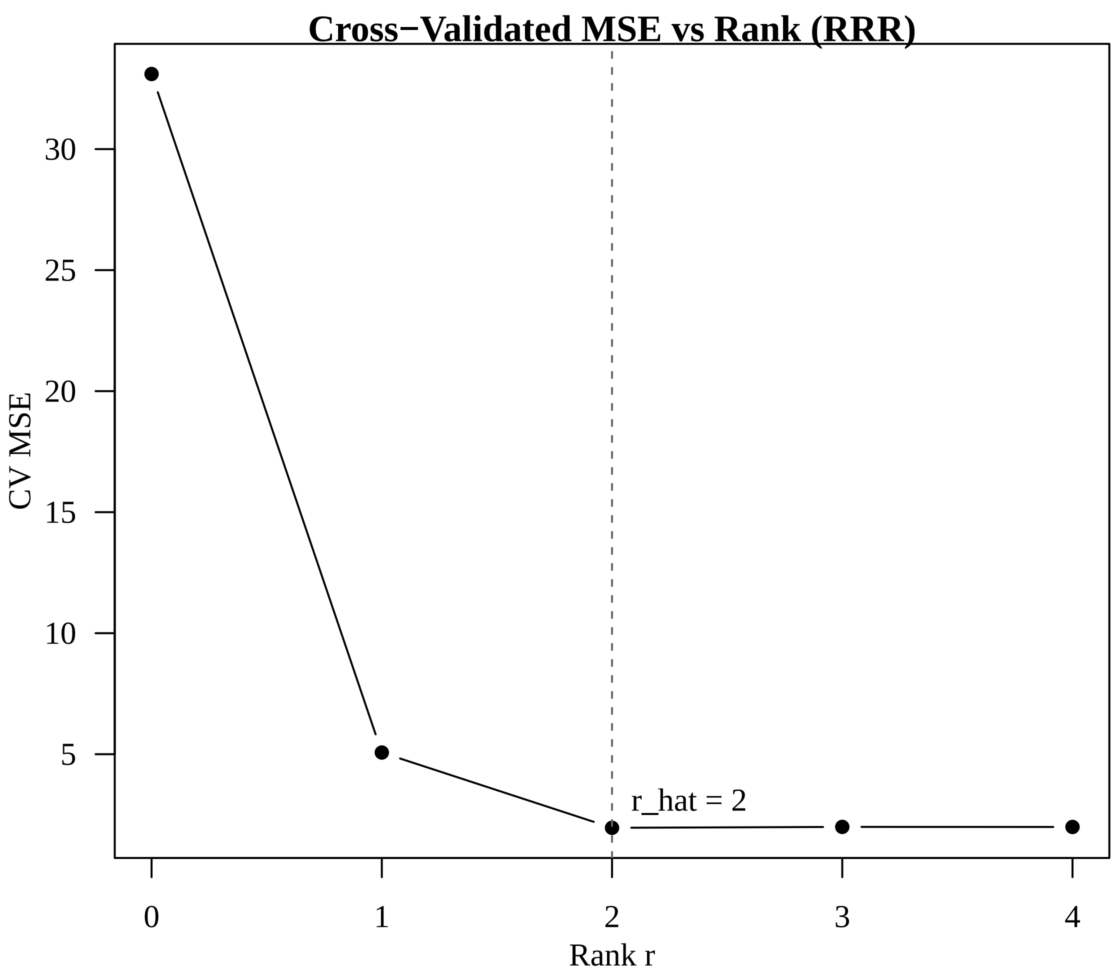
We evaluate predictive performance for candidate ranks using K-fold CV on rows (observations).

```r
cv_rrr <- function(X, Y, ranks, K = 5, seed = 1){
  set.seed(seed)
  n <- nrow(X)
  folds <- sample(rep(1:K, length.out = n))
  # store (r, fold, mse)
  res <- matrix(NA_real_, nrow = length(ranks) * K, ncol = 3)
  colnames(res) <- c("r","fold","mse")
  row_idx <- 1
  for(rr in ranks){
    for(k in 1:K){
      idx_te <- which(folds == k)
      idx_tr <- setdiff(seq_len(n), idx_te)
      Xtr <- X[idx_tr, , drop = FALSE]
      Ytr <- Y[idx_tr, , drop = FALSE]
      Xte <- X[idx_te, , drop = FALSE]
      Yte <- Y[idx_te, , drop = FALSE]
      if(rr == 0){
        B <- matrix(0, nrow = ncol(X), ncol = ncol(Y))
      } else {
        B <- RRR_fit(Xtr, Ytr, rr)$B_rrr
      }
      Ypred <- Xte %*% B
      mse <- mean((Yte - Ypred)^2)
      res[row_idx, ] <- c(rr, k, mse)
      row_idx <- row_idx + 1
    }
  }
  return(res)
}

ranks <- 0:min(ncol(X), ncol(Y))
cv_res <- cv_rrr(X, Y, ranks = ranks, K = 5, seed = 123)

# summarize by rank
cv_summary <- matrix(NA_real_, nrow = length(ranks), ncol = 2)
colnames(cv_summary) <- c("r","cv_mse")
for(i in seq_along(ranks)){
  r <- ranks[i]
  cv_summary[i, 1] <- r
  cv_summary[i, 2] <- mean(cv_res[cv_res[,1]==r, 3])
}
cv_summary
```

```
##      r    cv_mse
## [1,] 0 33.102885
## [2,] 1  5.069399
## [3,] 2  1.958754
## [4,] 3  1.996958
## [5,] 4  1.994387
```

Plot CV curve and select $\hat{r}$:

```r
par(mar = c(3, 3.5, 0.8, 0.6), mgp = c(2, 1, 0), las = 1,
    family = "serif")
plot(cv_summary[,1], cv_summary[,2], type = "b", xlab = "Rank r",
     ylab = "CV MSE", main = "Cross-Validated MSE vs Rank (RRR)",
     pch = 16)
r_hat <- cv_summary[which.min(cv_summary[,2]), 1]
abline(v = r_hat, lty = 2, col = "gray40")
text(r_hat, min(cv_summary[,2]) + 1, labels = paste("r_hat =", r_hat), pos = 4)
```



```
r_hat
```

```
## r
## 2
```

Compare OLS vs. RRR (Predictive Performance)

Hold out a test set to compare OLS and RRR predictions.

```
set.seed(1234)
test_idx <- sample(1:n, size = round(0.3 * n))
train_idx <- setdiff(1:n, test_idx)

Xtr <- X[train_idx, , drop = FALSE]
Ytr <- Y[train_idx, , drop = FALSE]
```

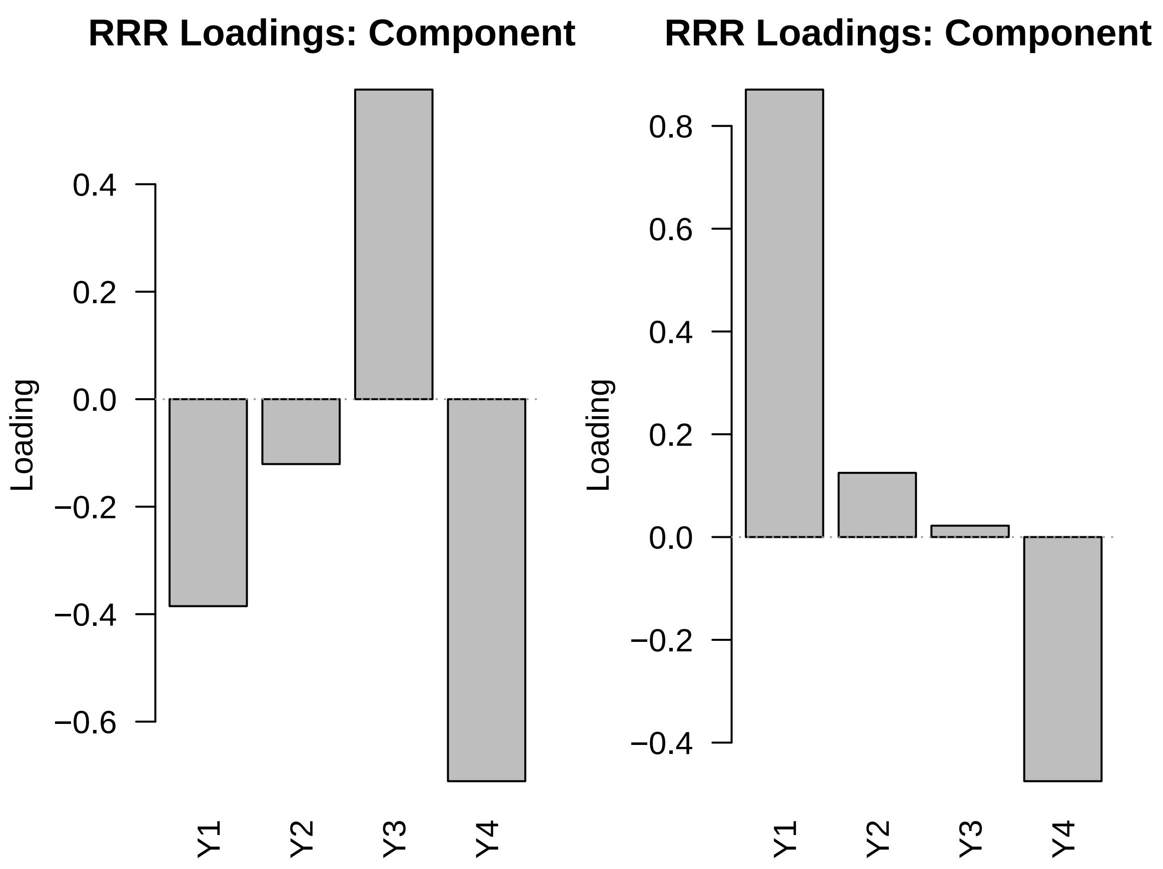
```r
Xte <- X[test_idx, , drop = FALSE]
Yte <- Y[test_idx, , drop = FALSE]

# OLS (multivariate): fit B_ols on train
B_ols_tr <- solve(crossprod(Xtr) + 1e-8 * diag(ncol(Xtr)), crossprod(Xtr, Ytr))
Yhat_ols <- Xte %*% B_ols_tr
mse_ols <- mean((Yte - Yhat_ols)^2)

# RRR (rank = r_hat) on train
r_use <- max(1, r_hat)  # ensure at least 1
B_rrr_tr <- RRR_fit(Xtr, Ytr, r = r_use)$B_rrr
Yhat_rrr <- Xte %*% B_rrr_tr
mse_rrr <- mean((Yte - Yhat_rrr)^2)

res_compare <- rbind(
  cbind(Method = "OLS (separate)", Test_MSE = mse_ols),
  cbind(Method = paste0("RRR (r=", r_use, ")"), Test_MSE = mse_rrr)
)
res_compare
```

```
##      Method           Test_MSE
## [1,] "OLS (separate)" "1.90507096441179"
## [2,] "RRR (r=2)"      "1.89022281117133"
```

Inspect Low-Rank Structure

Visualize the right singular vectors (response-space directions) that RRR uses.

```r
fit_full <- RRR_fit(X, Y, r = r_use)
Vr <- fit_full$Vr  # q x r_use

# Barplot of loadings by response for each component
op <- par(mfrow = c(1, max(1, ncol(Vr))), mar = c(4, 4, 3, 1))
for(j in 1:ncol(Vr)){
  barplot(Vr[, j],
          names.arg = colnames(Y),
          las = 2,
          main = paste("RRR Loadings: Component", j),
          ylab = "Loading")
  abline(h = 0, col = "gray60", lty = 3)
}
```



```r
par(op)
```

Repeated Measures Analysis

Dog Experiment

```r
dat <- read.table("dog1.txt")
temp <- array(dim = c(144, 4))
temp[, 1] <- rep(dat$V1, 4)
temp[, 2] <- rep(dat$V2, 4)
temp[, 3] <- rep(c(1, 5, 9, 13), each = 36)
temp[, 4] <- c(dat$V3, dat$V4, dat$V5, dat$V6)
dat2 <- data.frame(temp)
names(dat2) <- c("Treatment", "Dog_id", "Time", "Response")
dat2$Treatment <- as.factor(dat2$Treatment)
dat2$Dog_id <- as.factor(dat2$Dog_id)
dat2$Time <- as.factor(dat2$Time)
```

Split-plot ANOVA

```r
# computing the cell means (by treatment and time combinations)
tapply(dat2$Response, list(dat2$Treatment, dat2$Time), mean)
# interaction plot
```

```
par(las = 1, mgp = c(2.2, 1, 0), mar = c(3.6, 3.6, 0.8, 0.6))
with(dat2, interaction.plot(x.factor = Time, trace.factor = Treatment,
                            response = Response, col = 1:4, lwd = 1.5))
# Split-plot anova
library(lmerTest)
fit <- lmer(Response ~ Treatment * Time + (1 | Dog_id), data = dat2)
fit
anova(fit)
```

MANOVA

```
out <- manova(cbind(V3, V4, V5, V6) ~ as.factor(V1), data = dat)
summary(out, test = "Wilks")
```

```
##                Df   Wilks approx F num Df den Df  Pr(>F)
## as.factor(V1)  3 0.48452    2.022     12 77.018 0.03316 *
## Residuals     32
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(out)
```

```
##                Df Pillai approx F num Df den Df  Pr(>F)
## as.factor(V1)  3 0.5978   1.9286     12     93 0.04048 *
## Residuals     32
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Mixed Model with AR(1) temporal correlation structure

```
library(nlme)
fit1 = gls(Response ~ Treatment * Time,
           correlation = corCompSymm(form = ~ 1 | Dog_id), data = dat2)
fit1
```

```
## Generalized least squares fit by REML
##   Model: Response ~ Treatment * Time
##   Data: dat2
##   Log-restricted-likelihood: -119.9032
##
## Coefficients:
##       (Intercept)        Treatment2        Treatment3        Treatment4
##        4.11111111       -0.51111111       -0.46666667       -0.57111111
##             Time5             Time9            Time13 Treatment2:Time5
##        0.28888889        0.95555556        0.61111111        0.31111111
##  Treatment3:Time5  Treatment4:Time5  Treatment2:Time9  Treatment3:Time9
##        0.07777778       -0.20888889       -0.05555556       -0.62222222
##  Treatment4:Time9 Treatment2:Time13 Treatment3:Time13 Treatment4:Time13
##       -0.83555556        0.01388889       -0.21111111       -0.69111111
```

```
##
## Correlation Structure: Compound symmetry
##  Formula: ~1 | Dog_id
##  Parameter estimate(s):
##       Rho
## 0.5538616
## Degrees of freedom: 144 total; 128 residual
## Residual standard error: 0.6446676
```

```
fit2 = gls(Response ~ Treatment * Time,
           correlation = corAR1(form = ~ 1 | Dog_id), data = dat2)
fit2
```

```
## Generalized least squares fit by REML
##   Model: Response ~ Treatment * Time
##   Data: dat2
##   Log-restricted-likelihood: -120.7906
##
## Coefficients:
##       (Intercept)        Treatment2        Treatment3        Treatment4
##        4.11111111       -0.51111111       -0.46666667       -0.57111111
##             Time5             Time9            Time13  Treatment2:Time5
##        0.28888889        0.95555556        0.61111111        0.31111111
##  Treatment3:Time5  Treatment4:Time5  Treatment2:Time9  Treatment3:Time9
##        0.07777778       -0.20888889       -0.05555556       -0.62222222
##  Treatment4:Time9 Treatment2:Time13 Treatment3:Time13 Treatment4:Time13
##       -0.83555556        0.01388889       -0.21111111       -0.69111111
##
## Correlation Structure: AR(1)
##  Formula: ~1 | Dog_id
##  Parameter estimate(s):
##       Phi
## 0.5928708
## Degrees of freedom: 144 total; 128 residual
## Residual standard error: 0.6376364
```

```
anova(fit1, fit2)
```

```
##      Model df      AIC      BIC    logLik
## fit1     1 18 275.8063 327.1429 -119.9032
## fit2     2 18 277.5811 328.9177 -120.7906
```