STAT 528 - Advanced Regression Analysis II

Multivariate Regression

Daniel J. Eck
Department of Statistics
University of Illinois

Learning Objectives Today

- Multivariate regression modeling
- Examples

Motor Trend Cars example

The data was extracted from the 1974 Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973-74 models). The variables are:

- mpg: Miles/(US) gallon
- disp: Displacement (cu.in.)
- hp: Gross horsepower
- wt: Weight (1000 lbs)
- cyl: Number of cylinders
- am: Transmission (0 = automatic, 1 = manual)
- carb: Number of carburetors

The first four variables are response variables corresponding to engine performance and size. The next three variables are engine design variables.

We load in the data

```
data(mtcars)
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

The standard `lm` function in R can fit multivariate linear regression models.

```
mtcars$cyl <- factor(mtcars$cyl)
Y <- as.matrix(mtcars[,c("mpg","disp","hp","wt")])
m <- lm(Y ~ cyl + am + carb, data=mtcars, x = TRUE)

# estimate of beta'
betahat <- coef(m)
betahat
```

```
##                   mpg      disp         hp         wt
## (Intercept) 25.320303 134.32487 46.5201421  2.7612069
## cyl6        -3.549419  61.84324  0.9116288  0.1957229
## cyl8        -6.904637 218.99063 87.5910956  0.7723077
## am           4.226774 -43.80256  4.4472569 -1.0254749
## carb        -1.119855   1.72629 21.2764930  0.1749132
```

We now estimate Σ via MLE and provide an unbiased estimator.

```
# estimates of Sigma
SSE <- crossprod(Y - m$fitted.values)
n <- nrow(Y)
p <- nrow(coef(m))
SigmaMLE <- SSE / n
SigmaMLE
```

```
##              mpg       disp          hp         wt
## mpg    6.638633  -44.94796 -16.6232233 -0.5548030
## disp -44.947964 2113.48487 358.7058833 15.2773945
## hp   -16.623223  358.70588 487.0718440  0.3933977
## wt    -0.554803   15.27739   0.3933977  0.2171394
```

```
Sigmahat <- SSE / (n - p)
Sigmahat
```

```
##               mpg       disp          hp          wt
## mpg    7.8680094  -53.27166 -19.7015979 -0.6575443
## disp -53.2716607 2504.87095 425.1328988 18.1065416
## hp   -19.7015979  425.13290 577.2703337  0.4662491
## wt    -0.6575443   18.10654   0.4662491  0.2573503
```

We can see that the R's `vcov` function provides an estimate of $\mathrm{Var}(\mathrm{vec}(\hat{\beta}')) = \Sigma \otimes (\mathbb{X}'\mathbb{X})^{-1}$ as its default

```r
X <- m$x
unique(round(vcov(m) - kronecker(Sigmahat, solve(crossprod(X))), 10))
```

```
##                 mpg:(Intercept) mpg:cyl6 mpg:cyl8 mpg:am mpg:carb
## mpg:(Intercept)               0        0        0      0        0
##                 disp:(Intercept) disp:cyl6 disp:cyl8 disp:am disp:carb
## mpg:(Intercept)                0         0         0       0         0
##                 hp:(Intercept) hp:cyl6 hp:cyl8 hp:am hp:carb wt:(Intercept)
## mpg:(Intercept)              0       0       0     0       0              0
##                 wt:cyl6 wt:cyl8 wt:am wt:carb
## mpg:(Intercept)       0       0     0       0
```

We obtain inferences for regression coefficients corresponding to the first response variable `mpg`.

```r
# summary table from lm
msum <- summary(m)
msum[[1]]
```

```
##
## Call:
## lm(formula = mpg ~ cyl + am + carb, data = mtcars, x = TRUE)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9074 -1.1723  0.2538  1.4851  5.4728
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  25.3203     1.2238  20.690  < 2e-16 ***
## cyl6         -3.5494     1.7296  -2.052 0.049959 *
## cyl8         -6.9046     1.8078  -3.819 0.000712 ***
## am            4.2268     1.3499   3.131 0.004156 **
## carb         -1.1199     0.4354  -2.572 0.015923 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.805 on 27 degrees of freedom
## Multiple R-squared:  0.8113, Adjusted R-squared:  0.7834
## F-statistic: 29.03 on 4 and 27 DF,  p-value: 1.991e-09
```

We compare the summary table on the previous design to one obtained directly from theory

```r
## estimates and standard errors
msum2 <- cbind(coef(m)[, 1], sqrt(diag( kronecker(Sigmahat, solve(crossprod(X))) ))[1:5])

## tstat and p-values
msum2 <- cbind(msum2, msum2[, 1] / msum2[, 2])
msum2 <- cbind(msum2, sapply(msum2[, 3], function(x) pt(abs(x), df = n - p, lower = FALSE)*2 ))
msum2
```

```
##                  [,1]      [,2]      [,3]         [,4]
## (Intercept) 25.320303 1.2237903 20.690067 4.303483e-18
## cyl6        -3.549419 1.7295506 -2.052221 4.995947e-02
## cyl8        -6.904637 1.8078219 -3.819313 7.124146e-04
## am           4.226774 1.3499249  3.131118 4.156214e-03
## carb        -1.119855 0.4353558 -2.572274 1.592280e-02
```

Inference and nested models

There are a plethora of additional tests that we could perform in this setting (we will not stress each test's origins in this course).

These include the Wilk's lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root.

First, let $E = n\tilde{\Sigma}$ where $\tilde{\Sigma}$ is the MLE for the full model with β unconstrained and let $\tilde{H} = n(\tilde{\Sigma}_1 - \tilde{\Sigma})$ where $\tilde{\Sigma}_1$ is the MLE of Σ in the reduced model constrained by $\beta_2 = 0$.

The test statistics now follow:

- Wilk's lambda $= \prod_{i=1}^{s} \frac{1}{1+\eta_i} = \frac{|\tilde{E}|}{|\tilde{E}+\tilde{H}|}$
- Pillai's trace $= \sum_{i=1}^{s} \frac{\eta_i}{1+\eta_i} = \mathrm{tr}[\tilde{H}(\tilde{E} + \tilde{H})^{-1}]$
- Hotelling-Lawley trace $= \sum_{i=1}^{s} \eta_i = \mathrm{tr}(\tilde{H}\tilde{E}^{-1})$
- Roy's greatest root $= \frac{\eta_1}{1+\eta_1}$

where $\eta_1 \geq \eta_2 \geq \cdots \geq \eta_s$ denote the nonzero eigenvalues of $\tilde{H}\tilde{E}^{-1}$.

We demonstrate estimation of Wilk's lambda and Pillai's trace.

```
## anova implements these methods
m0 <- lm(Y ~ am + carb, data=mtcars)
anova(m0, m, test="Wilks")
```

```
## Analysis of Variance Table
##
## Model 1: Y ~ am + carb
## Model 2: Y ~ cyl + am + carb
##   Res.Df Df Gen.var.   Wilks approx F num Df den Df    Pr(>F)
## 1     29      43.692
## 2     27 -2   29.862 0.16395   8.8181      8     48 2.525e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(m0, m, test="Pillai")
```

```
## Analysis of Variance Table
##
## Model 1: Y ~ am + carb
## Model 2: Y ~ cyl + am + carb
##   Res.Df Df Gen.var. Pillai approx F num Df den Df    Pr(>F)
## 1     29      43.692
## 2     27 -2   29.862 1.0323   6.6672      8     50 6.593e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The same quantities are estimated directly

```
Etilde <- n * SigmaMLE
SigmaTilde1 <- crossprod(Y - m0$fitted.values) / n
Htilde <- n * (SigmaTilde1 - SigmaMLE)
HEi <- Htilde %*% solve(Etilde)
HEi.values <- eigen(HEi)$values
c(Wilks = prod(1 / (1 + HEi.values)), Pillai = sum(HEi.values / (1 + HEi.values)))
```

```
##        Wilks       Pillai
## 0.1639527+0i 1.0322975+0i
```

We will now demonstrate confidence and prediction intervals in the motor trends cars example. Note that we have to code our own functions because R does not have the capability to provide these quantities. R does provide point predictions.

```
# confidence interval
newdata <- data.frame(cyl=factor(6, levels=c(4,6,8)), am=1, carb=4)
predict(m, newdata, interval="confidence")
```

```
##        mpg     disp      hp       wt
## 1 21.51824 159.2707 136.985 2.631108
```

```
# prediction interval
newdata <- data.frame(cyl=factor(6, levels=c(4,6,8)), am=1, carb=4)
predict(m, newdata, interval="prediction")
```

```
##        mpg     disp      hp       wt
## 1 21.51824 159.2707 136.985 2.631108
```

Here is the function which produces confidence and prediction intervals (credit to Nathaniel Helwig)

We now provide 95% confidence and prediction intervals using code from Nathaniel Helwig (see Rmd file):

```
# confidence interval
newdata <- data.frame(cyl=factor(6, levels=c(4,6,8)), am=1, carb=4)
pred.mlm(m, newdata)
```

```
##          mpg     disp        hp       wt
## fit 21.51824 159.2707 136.98500 2.631108
## lwr 16.65593  72.5141  95.33649 1.751736
## upr 26.38055 246.0273 178.63351 3.510479
```

```
# prediction interval
newdata <- data.frame(cyl=factor(6, levels=c(4,6,8)), am=1, carb=4)
pred.mlm(m, newdata, interval="prediction")
```

```
##           mpg      disp        hp        wt
## fit 21.518240 159.27070 136.98500 2.6311076
## lwr  9.680053 -51.95435  35.58397 0.4901152
## upr 33.356426 370.49576 238.38603 4.7720999
```