QR Execrcise 3.5 MATH531

Jared Andreatta

2025-01-28

Getting started

First here is an example dataset that you can use to illustrate OLS computations. Loading the AudiA4 data.The goal is to be able to predict the price of a used A4 based on the mileage and model year.

```
library( fields) # load fields package
```

```
## Warning: package 'fields' was built under R version 4.4.2

## Loading required package: spam

## Warning: package 'spam' was built under R version 4.4.2

## Spam version 2.11-1 (2025-01-20) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

##
## Attaching package: 'spam'

## The following objects are masked from 'package:base':
##
##     backsolve, forwardsolve

## Loading required package: viridisLite

##
## Try help(fields) to get started.
```

```
load("AudiA4.rda" )
head( AudiA4)
```

```
##     year price mileage distance
## 58  2020 39649    3848       29
## 145 2020 43175    3962        7
## 10  2020 43675    5316        7
## 52  2020 40649    5417       29
## 143 2020 42175    5846        7
## 9   2020 45675    6539        7
```

```
# scale and adjust variables to be easier to interpret.
year<- AudiA4$year- min(AudiA4$year)
mileage<- AudiA4$mileage/1000
Y<- AudiA4$price/1000

X<- cbind( 1, mileage, year  )

fit<-   lm( Y ~ X - 1)
```

For those students rusty with linear algebra in R here is a digression to find OLS estimates found "by hand".

```
betaHat<- solve(t(X)%*%X)%*%t(X)%*%Y
# check
bothBetas<- cbind(betaHat,fit$coefficients  )
print( bothBetas)
```

```
##                [,1]       [,2]
##          14.3751275 14.3751275
## mileage -0.1241398 -0.1241398
## year      1.1550828  1.1550828
```

```
XbetaHat<- X%*%betaHat
test.for.zero(XbetaHat, fit$fitted.values )
```

```
## PASSED test at tolerance  1e-08
```

Problem 1

- Define the QR decomposition of an $n \times p$ matrix X.
  Distinguish between the *skinny* QR where Q is $n \times p$ and R is $p \times p$ and the *fat* one where where Q is $n \times n$ and R is $n \times p$. (The skinny one is usually what is computed.)
  - The "skinny" QR Decomposition refers to the situation in which $Q \in \mathbb{R}^{n \times p}$ where Q is comprised of orthonormal column vectors, and $n > p$, and more often than not, $n >> p$, as there tends to be more observations than features in the dataset; $R \in \mathbb{R}^{n \times n}$ is an upper right triangular matrix. In this case, the columns of X are expressed as a linear combination with the orthonormal columns of Q.
    The "fat" QR Decomposition, where $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{n \times p}$. The difference in this is the number of columns in Q; Q has n columns that form an orthonormal basis of $\mathbb{R}^n$. R is a rectangular matrix of dimension $n \times p$, where the upper triangular block has dimension of $p \times p$.
    To summarize, the "skinny" decomposition is often used for "tall" matrices due to its computational properties, whereas the "fat" decomposition is used to form an orthonormal basis that spans $\mathbb{R}^n$.
- Who invented the QR decomposition?
  - The QR algorithm was first proposed by John G. F. Francis and Vera N. Kublanovskaya (not collaboratively) in the late 1950s.
- Using the function `qr` and related functions (`qr.coef`, `qr.Q`, etc.) in R find the skinny QR decomposition of X (List the first 5 rows of "Q" and all of "R")

```r
q = qr(X)
head(qr.Q(q))
```

```
##             [,1]       [,2]         [,3]
## [1,] -0.09166985 -0.1191055 -0.005470635
## [2,] -0.09166985 -0.1188379 -0.005921018
## [3,] -0.09166985 -0.1156595 -0.011270299
## [4,] -0.09166985 -0.1154224 -0.011669323
## [5,] -0.09166985 -0.1144153 -0.013364184
## [6,] -0.09166985 -0.1127886 -0.016102036
```

```r
print(qr.R(q))
```

```
##                  mileage     year
## [1,] -10.90871 -595.4751 -137.7798
## [2,]   0.00000  426.0015  -35.7134
## [3,]   0.00000    0.0000  -21.2199
```

Problem 2

Use the QR decomposition to compute the OLS estimate for the Audi A4 data and verify that it is same as the `lm` result above.

```r
QR_X <- qr.Q(q)%*%qr.R(q)
QRbetaHat <- QR_X%*%betaHat
test.for.zero(XbetaHat, fit$fitted.values )
```

```
## PASSED test at tolerance  1e-08
```

```r
head(QRbetaHat)
```

```
##          [,1]
## [1,] 33.53385
## [2,] 33.51969
## [3,] 33.35161
## [4,] 33.33907
## [5,] 33.28581
## [6,] 33.19979
```

```r
head(XbetaHat)
```

```
##          [,1]
## [1,] 33.53385
## [2,] 33.51969
## [3,] 33.35161
## [4,] 33.33907
## [5,] 33.28581
## [6,] 33.19979
```

Both estimators give the same output