STAT 516 Lec 04

Multiple linear regression (part 2/2)

Karl Gregory

2025-01-28

Rental rates of commercial properties example

As in part $1/2$, consider these data from Kutner et al. (2005).

```
link <- url("https://people.stat.sc.edu/gregorkb/data/KNLIcp.txt")
commprop <- read.table(link,col.names=c("rent","age","optx","vac","sqft"))
commprop$sqft <- commprop$sqft/10000 # rescale sqft
head(commprop)
```

```
  rent age  optx  vac    sqft
1 13.5   1  5.02 0.14 12.3000
2 12.0  14  8.19 0.27 10.4079
3 10.5  16  3.00 0.00  3.9998
4 15.0   4 10.70 0.05  5.7112
5 14.0  11  8.97 0.07  6.0000
6 10.5  15  9.45 0.24 10.1385
```

```
n <- nrow(commprop)
p <- ncol(commprop) - 1
```

There are $n = 81$ rows and $p = 4$ predictors.

Setup

Consider data $(Y_1, \mathbf{x}_1), \dots, (Y_n, \mathbf{x}_n)$, with each $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$.

The multiple linear regression model is

$$Y_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ip}\beta_p + \varepsilon_i, \quad i = 1, \dots, n,$$

where

- $\mathbf{x}_1, \dots, \mathbf{x}_n$ are vectors in $\mathbb{R}^p$ of covariate or predictor values.
- $Y_1, \dots, Y_n$ are the response values
- $\beta_0, \beta_1, \dots, \beta_p$ are the regression coefficients.
- $\varepsilon_1, \dots, \varepsilon_n$ are iid Normal$(0, \sigma^2)$ error terms.
- σ^2 is the error term variance.

Goals in multiple linear regression

In part 1/2, we addressed these goals:

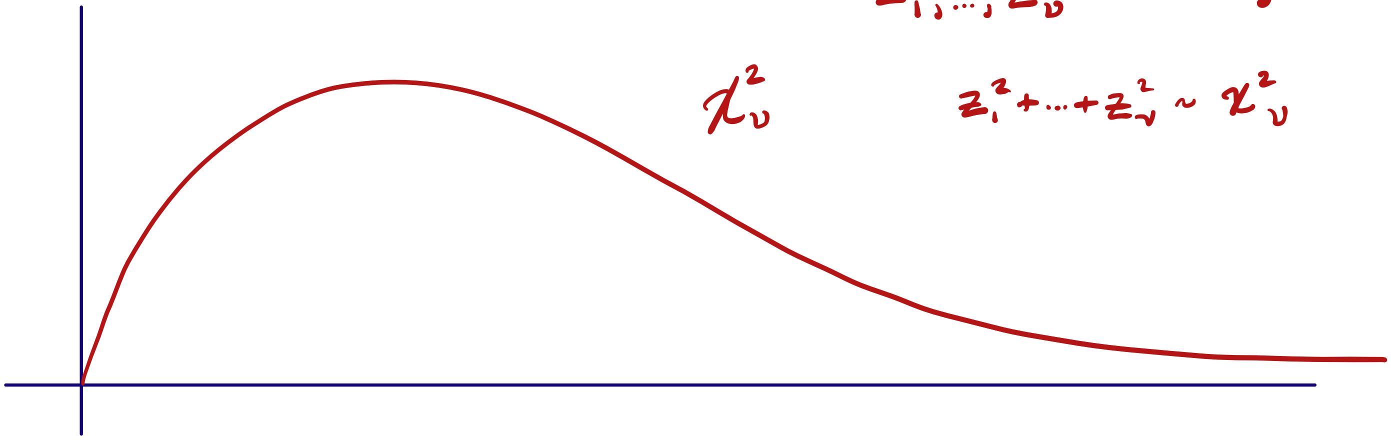
1. Estimate the regression coefficients β_0 and $\beta_1, \dots, \beta_p$.
2. Estimate the error term variance σ^2.
3. Perform inference on $\beta_1, \dots, \beta_p$.
4. Build a CI for $\beta_0 + \beta_1 x_{\text{new},1} + \cdots + \beta_p x_{\text{new},p}$ at any $\mathbf{x}_{\text{new}}$.
5. Build a prediction interval for Y at any $\mathbf{x}_{\text{new}}$.
6. Decompose the variation in Y into (sums of) sums of squares.
7. Check whether the model assumptions are satisfied.
8. Identify outliers and understand their effects.

In part 2/2 we focus on these:

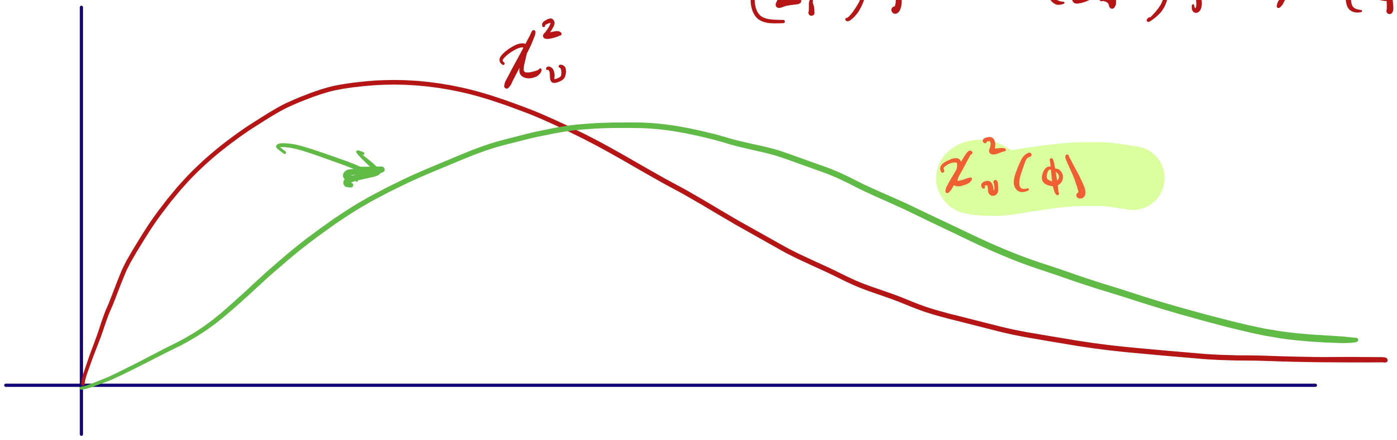
8. Test for significance of a subset of covariates
9. Understand how correlations among the covariates affect inferences
10. Do variable selection

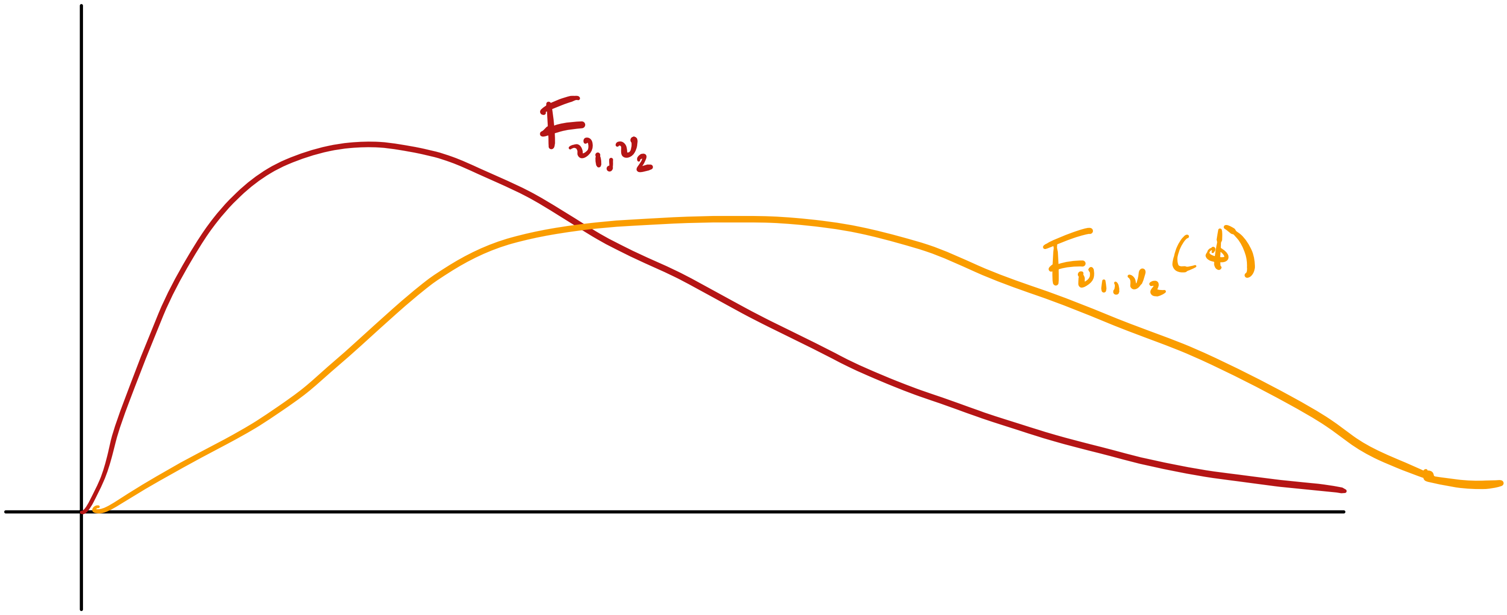
$Z_1, \dots, Z_\nu \overset{iid}{\sim} N(0,1)$

$Z_1^2 + \dots + Z_\nu^2 \sim \chi^2_\nu$



$(Z_1 - \mu)^2 + \dots + (Z_\nu - \mu)^2 \sim \chi^2(\phi = \mu)$





Review of F distributions

"phi": noncentrality parameter

"Ratio"

For $\underline{W_1} \sim \chi^2_{\nu_1}(\phi)$, $\underline{W_2} \sim \chi^2_{\underline{\nu_2}}$ independent, $R = \frac{\boxed{W_1/\nu_1}}{\boxed{W_2/\nu_2}} \sim F_{\nu_1,\nu_2}(\phi)$.

"nu"

$F_{\nu_1,\nu_2}(\phi)$ denotes the F distribution with

- numerator degrees of freedom ν_1
- denominator degrees of freedom ν_2
- noncentrality parameter $\phi \geq 0$

If $\phi > 0$ the distribution is a non-central F distribution.

When $\phi = 0$ we just write F_{ν_1,ν_2} to denote the "central" F distribution.

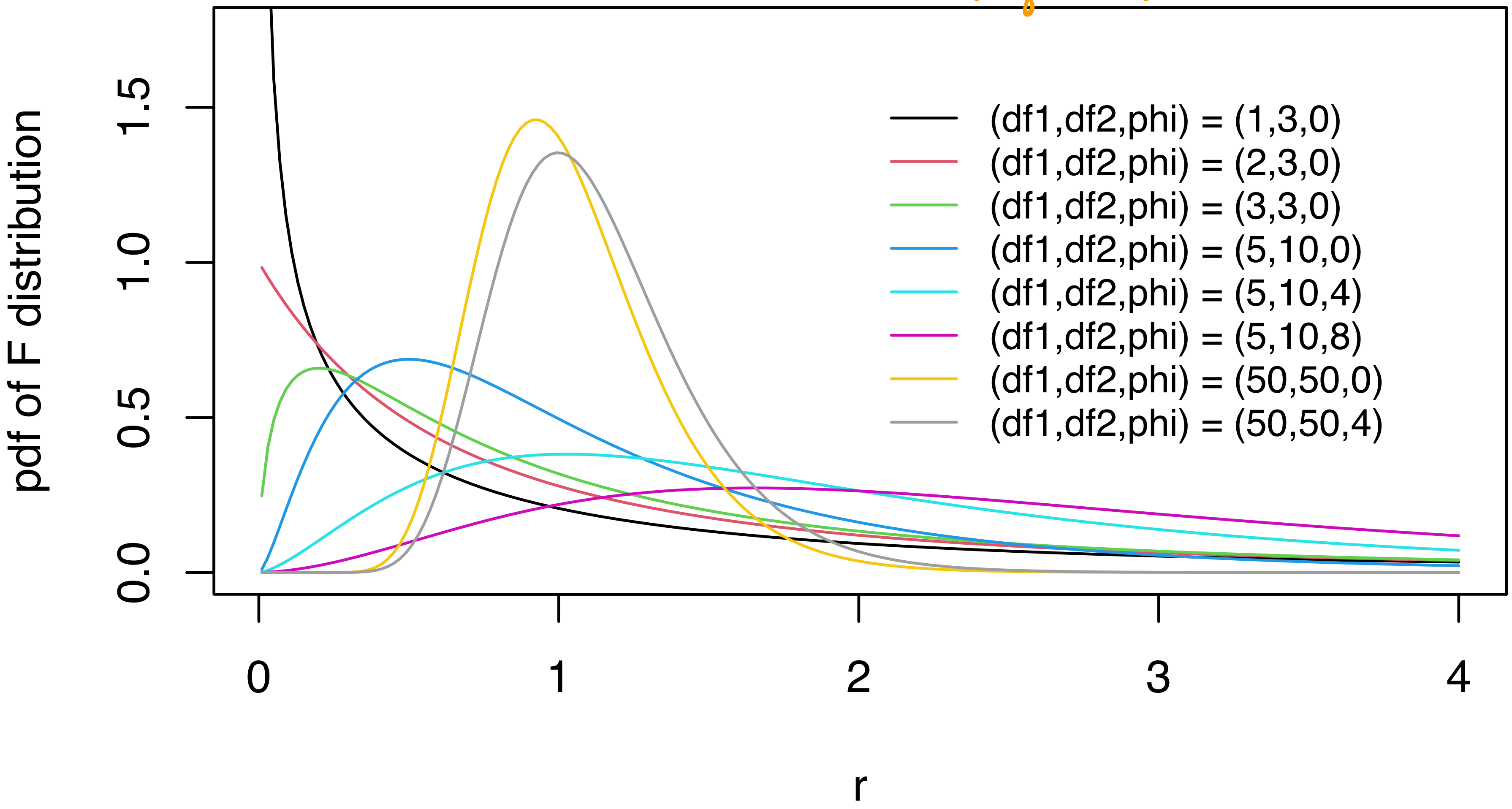
We will encounter ratios of sums of squares which have F distributions.

Plot of some F distribution pdfs

```
nu1 <- c(1,2,3,5,5,5,50,50)
nu2 <- c(3,3,3,10,10,10,50,50)
phi <- c(0,0,0,0,4,8,0,4)
f <- seq(.01,4,length=200)
dfmat <- matrix(0,length(f),200)
for(j in 1:length(nu1)){

  dfmat[j,] <- df(f,df1 = nu1[j],df2=nu2[j],ncp=phi[j])

}
lab <- paste("(df1,df2,phi) = (",
      apply(cbind(nu1,nu2,phi),1,paste,collapse = ","),
      ")",sep="")
```

```
plot(NA,xlim = range(f),ylim = c(0,1.2*max(dfmat[-1,])),
     xlab = "r",
     ylab = "pdf of F distribution")
for(j in 1:length(nu1)) lines(dfmat[j,]~f, col = j)
legend(x = .5*max(f),y = 1.1*max(dfmat[-1,]),legend = lab,
       col = 1:length(nu1), lty = 1,bty = "n", cex = .8)
```

Right-skewed.



The overall F-test

(Omnibus test)

$H_0: \beta_1 = 0, \beta_2 = 0, \dots \beta_p = 0$

$H_1:$ at least one $\beta_j \neq 0$.

We may wish to test whether *any* covariates are important, that is

$$H_0\colon \beta_j = 0 \text{ for all } j = 1, \dots, p.$$

The overall F-test of significance is carried out as follows:

1. Fit the model with all the covariates and obtain the value

$$F_{\text{stat}} = \frac{\text{MS}_{\text{Reg}}}{\text{MS}_{\text{Error}}} \left(= \underbrace{\frac{\text{SS}_{\text{Reg}}/p}{\text{SS}_{\text{Error}}/(n-(p+1))}}_{\frac{\chi^2_p/p}{\chi^2_{n-(p+1)}/(n-(p+1))}} \right) \overset{\text{if } H_0 \text{ true}}{\sim} F_{p,n-(p+1)}$$

2. Reject H_0 at α if $F_{\text{stat}} > F_{p,n-(p+1),\alpha}$.
3. Obtain p-value is $P(F > F_{\text{stat}})$, where $F \sim F_{p,n-(p+1)}$.

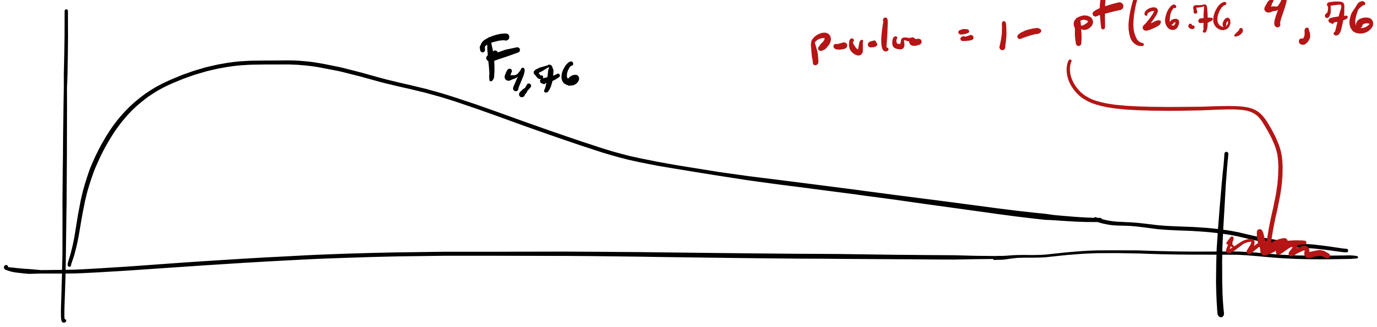
This test statistic and p-value are reported by `summary()` on `lm()`.

On Data:

$H_0: \beta_j = 0$ for all $j = 1, \dots, 4$

$F_{stat} = 26.76$ $\quad$ $df_{num} = p = 4$

$df_{den} = n - (p+1) = 81 - (4+1) = 76$

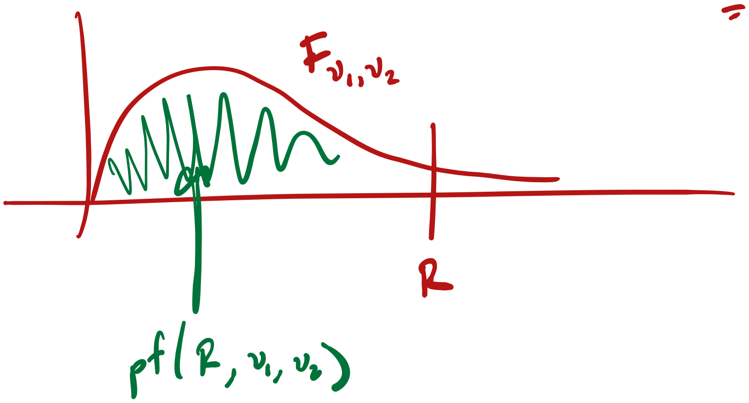


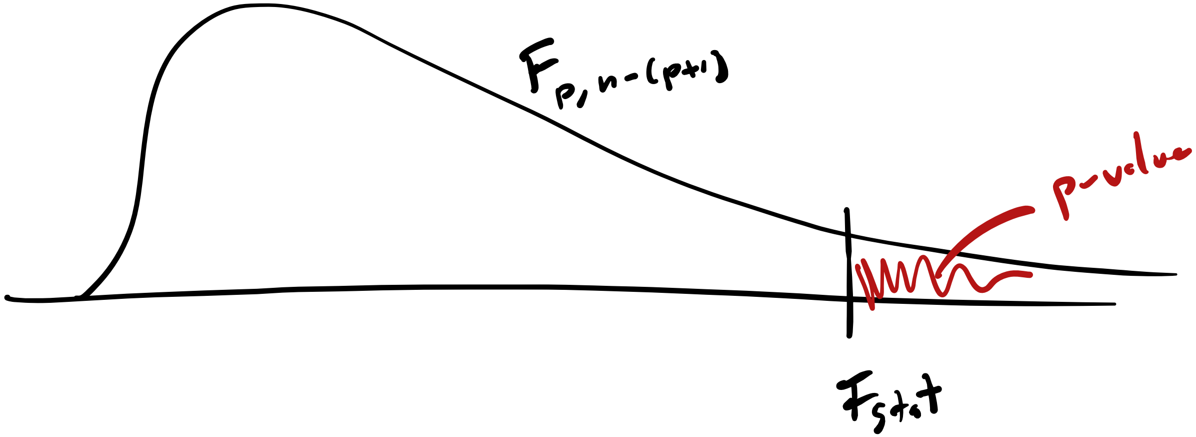
p-value $= 1 - pf(26.76, 4, 76)$

$F_{stat} = 26.76$

p-value $= 7.2 \times 10^{-14}$

$\Rightarrow$ Reject H_0.





Exercise: Show that the test statistic of the overall F test can be written

$$F_{\text{stat}} = \frac{\text{MS}_{\text{Reg}}}{\text{MS}_{\text{Error}}} = \underline{\frac{(n-(p+1))}{p}\frac{R^2}{1-R^2}},$$

where R^2 is the coefficient of determination.

$$R^2 = \frac{SS_{Reg}}{SS_{Tot}}$$

RHS: $$\frac{n-(p+1)}{p} * \frac{\frac{SS_{Reg}}{SS_{Total}}}{1-\frac{SS_{Reg}}{SS_{Total}}} = \frac{n-(p+1)}{p}\frac{\frac{SS_{Reg}}{\cancel{SS_{Total}}}}{\frac{SS_{Error}}{\cancel{SS_{Total}}}} = \frac{\frac{SS_{Reg}}{p}}{\frac{SS_{Error}}{n-(p+1)}} = \frac{MS_{Reg}}{MS_{Error}} = F_{stat}$$

$$SS_{Total} = SS_{Reg} + SS_{Error} \longrightarrow \frac{SS_{Total} - SS_{Reg}}{SS_{Tot}} = \frac{SS_{Error}}{SS_{Tot}}$$

Exercise: Suppose you fit a regression model with 3 predictors on a data set with 81 observations, and you obtain $\hat{\sigma} = 1.132$ and $R^2 = 0.583$. Use this information to fill in the entire ANOVA table:

Source	Df	SS	MS	F value	p-value
Regression	p 3	SS_{Reg}	MS_{Reg}	F_{stat}	$P(F > F_{\text{stat}})$
Error	$n-(p+1)$ 77	SS_{Error}	MS_{Error}		
Total	$n-1$ 80	SS_{Tot}			

$p = 3$

$n = 81$

$\hat{\sigma} = 1.132$

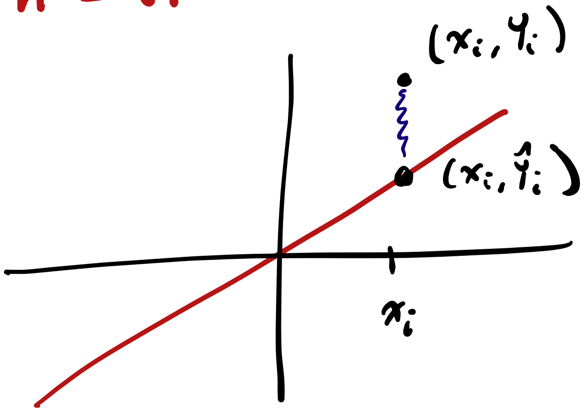
$$\hat{\sigma}^2 = \frac{1}{n-(p+1)} \sum_{i=1}^{n} \hat{\varepsilon}_i^2$$

$$= \frac{1}{n-(p+1)} \underbrace{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}_{SSE}$$

$$= MS_{Error}$$

$$MS_{Error} = (1.132)^2$$

$$= 1.281$$



$$R^2 = \frac{SS_{Reg}}{SS_{Tot}} \iff SS_{Total} = \frac{SS_{Reg}}{R^2}$$

Source	Df	SS	MS	F	p-val
Reg	3	137.90	45.97	35.81	
Error	77	98.637	1.281		1-pf(35.81, 3, 77) ≈ 0
Total	80	236.54			

$$R^2 = 0.583 = \frac{SS_{Reg}}{SS_{Total}} = \frac{SS_{Total} - SS_{Error}}{SS_{Total}}$$

$$= 1 - \frac{SS_{Error}}{SS_{Total}}$$

$$\iff 1 - R^2 = \frac{SS_{Error}}{SS_{Total}}$$

$$SS_{Total} = \frac{SS_{Error}}{1 - R^2} = 236.54$$

$$SS_{Total} = SS_{Reg} + SS_{Error}$$

$$\iff SS_{Reg} = SS_{Tot} - SS_{Error}$$

Testing for significance of a subset of covariates

Consider testing the significance of a subset $D \subset \{1, \ldots, p\}$ of covariates:

$$H_0\colon\ \beta_j = 0 \text{ for all } j \in D.$$

Ex: $D = \{1, 2\}$

Use the full-reduced model F-test:

1. Let s be the number of covariates in D and compute

$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ from model without covariates in D

$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$ from model with All covariates

$$F_{\text{stat}} = \frac{(\text{SS}_{\text{Error}}(\text{Reduced}) - \text{SS}_{\text{Error}}(\text{Full}))/s}{\text{SS}_{\text{Error}}(\text{Full})/(\underbrace{n - (p+1)}_{\text{denom df}})},$$

numerator df

   - "Full" is the model with all p covariates.
   - "Reduced" is the model after dropping the covariates in D.

2. Reject H_0 at α if $F_{\text{stat}} > F_{s, n-(p+1), \alpha}$.
3. Obtain p-value as $P(F > F_{\text{stat}})$, where $F \sim F_{s, n-(p+1)}$.

Rent data

$$Rent_i = \beta_0 + \beta_1 * age_i + \beta_2 * optx_i + \beta_3 * vac_i + \beta_4 * sqft_i + \varepsilon_i$$

Example of Full- Reduced model F- test

H_0: $\beta_2 = 0$ and $\beta_3 = 0$

vs H_1 : H_0 not true

Fit two models.

Full:

$$Rent_i = \beta_0 + \beta_1 * age_i + \underline{\beta_2 * optx_i + \beta_3 * vac_i} + \beta_4 * sqft_i + \varepsilon_i$$

Get SS_{Error} (full) (sum of squared residuals) $= \sum_{i=1}^{n} \underbrace{(Y_i - \hat{Y}_i)^2}_{\text{residual}}$

$= \sum_{i=1}^{n} \hat{\varepsilon}_i^2$

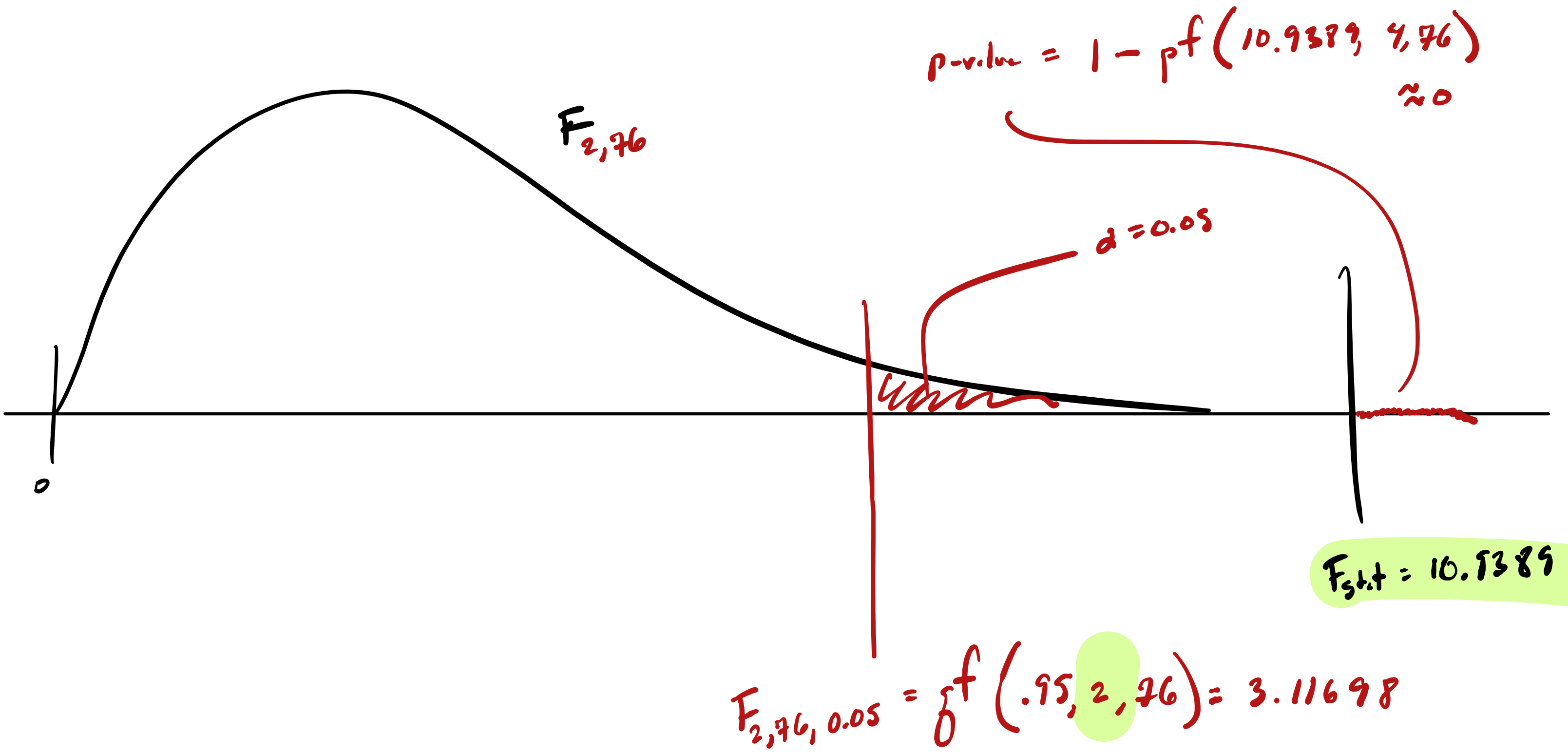
Reduced:

$$Rent_i = \beta_0 + \beta_1 * age_i + \beta_4 * sqft_i + \varepsilon_i$$

Get SS_{Error} (Red)

$$F_{stat} = \frac{\left(SS_{Error}(Red) - SS_{Error}(Full)\right) / \overset{\text{num df}}{2}}{SS_{Error}(Full) / \underbrace{(n-(p+1))}_{81-(4+1)=76}} = 10.9389$$

denom



So Reject $H_0: \beta_2 = 0$ and $\beta_3 = 0$.

Rental rates of commercial properties example (cont)

Check whether vac and optx contribute significantly to the rent.

That is test H_0: $\beta_{\text{vac}} = 0$ and $\beta_{\text{optx}} = 0$.

```r
lm_red <- lm(rent ~ age + sqft, data = commprop)
lm_full <- lm(rent ~ age + optx + vac + sqft, data = commprop)
SSE_red <- sum(lm_red$residuals^2)
SSE_full <- sum(lm_full$residuals^2)
s <- 2 # significance of two covariates being tested
Fstat <- (SSE_red - SSE_full)/s / ( SSE_full / (n - (p + 1)))
alpha <- 0.05
F_crit <- qf(1 - alpha,s,n-(p+1))
pval <- 1 - pf(Fstat, 2, n - (p+1))
```

We obtain $F_{\text{stat}} = 10.939$ and $F_{s,n-(p+1),0.05} = 3.117$, and the p-value is 0.

$$Rent_i = \beta_0 + \beta_1 * age_i + \beta_2 * optx_i + \beta_3 * vac_i + \beta_4 * sqft_i + \varepsilon_i$$

Full/Reduced for a single covariate:

$$H_0: \beta_3 = 0 \quad vs \quad H_1: \beta_3 \neq 0.$$

Full model:

$$Rent_i = \beta_0 + \beta_1 * age_i + \beta_2 * optx_i + \beta_3 * vac_i + \beta_4 * sqft_i + \varepsilon_i$$

Red model:

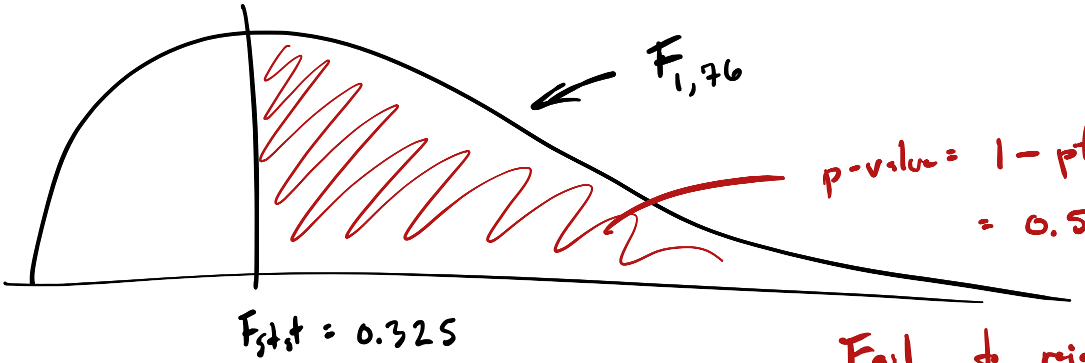
$$Rent_i = \beta_0 + \beta_1 * age_i + \beta_2 * optx_i + \beta_4 * sqft_i + \varepsilon_i$$

only removing 1 covariate

$$F_{stat} = \frac{\left(SS_{Err}(Red) - SS_{Err}(Full)\right) / 1}{SS_{Err}(Full) / \underbrace{(n-(p+1))}_{76}}$$

$$\vdots$$

$$= 0.32475$$



p-value $= 1 - pf(0.325, 1, 76)$

$= 0.570$

Fail to reject H_0.

Full-reduced model F test for a single covariate

If we test H_0: $\beta_j = 0$ for a single covariate using the full-reduced model F test, the test statistic F_{stat} will be equal to the square of the test statistic T_{stat} for testing H_0: $\beta_j = 0$ in the full model.

```
lm_red <- lm(rent ~ age + optx + sqft, data = commprop)
lm_full <- lm(rent ~ age + optx + vac + sqft, data = commprop)
SSE_red <- sum(lm_red$residuals^2)
SSE_full <- sum(lm_full$residuals^2)
s <- 1 # significance of a single covariate being tested
Fstat <- (SSE_red - SSE_full)/s / ( SSE_full / (n - (p + 1)))
sqrt(Fstat) # absolute value of the t-statistic from the full model
```

```
[1] 0.5698714
```

END EXAM I MATERIAL

$$Y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

Get estimators $\hat{\beta}_1, \dots, \hat{\beta}_p$

$$\Omega = \left(X^T X\right)^{-1}$$

$$\mathrm{Var}\ \hat{\beta}_j = \frac{\sigma^2 \Omega_{jj}}{n}$$

$$\vdots$$

$$= \frac{1}{1 - R_j^2} \frac{\sigma^2}{\sum_{i=1}^{n} (x_{ji} - \bar{x}_j)^2},$$

Measure how spread out values of X_j are.

R_j^2 = The R^2 you get if you regress X_j on the other covariates

If X_j is highly correlated with the other covariates we will get R_j^2 close to 1.

Then it becomes hard to estimate β_j.

Var $\hat{\beta}_1$ is large.

Effect of correlations among the covariates

From before $\operatorname{Var}\hat{\beta}_j = \sigma^2\Omega_{jj}/n$. An alternate expression gives

$$\operatorname{Var}\hat{\beta}_j = \frac{1}{1-R_j^2}\frac{\sigma^2}{\sum_{i=1}^n (x_{ji}-\bar{x}_j)^2},$$

where R_j^2 is the R^2 from regressing x_j on the other covariates.

So multicollinearity of x_j with the other covariates "inflates" $\operatorname{Var}\hat{\beta}_j$:

- Makes confidence intervals for β_j wider.
- Makes tests of H_0: $\beta_j = 0$ vs H_1: $\beta_j \neq 0$ less powerful.

Call $\dfrac{1}{1-R_j^2}$ the variance inflation factor (VIF) for x_j, $j = 1, \dots, p$.

VIFs in commercial properties example

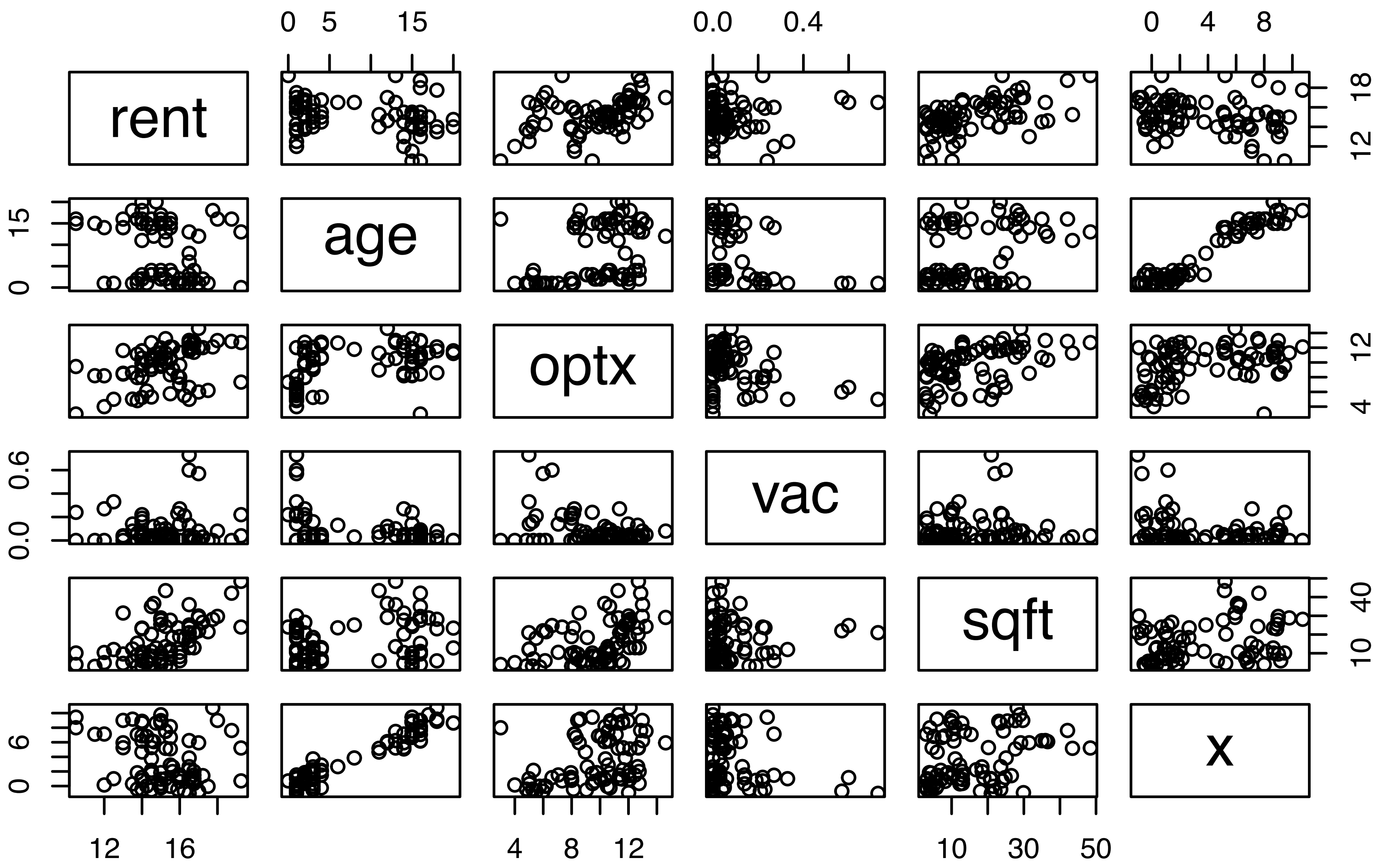
Add to the data set a spurious predictor highly correlated with age.

Check the effect of this on our inferences for β_{age}.

```
# make new x correlated with age
x <- .5*commprop$age + rnorm(n)
commpropx <- cbind(commprop,x)
round(cor(commpropx),4)
```

```
        rent     age    optx     vac   sqft       x
rent  1.0000 -0.2503  0.4138  0.0665 0.5353 -0.2354
age  -0.2503  1.0000  0.3888 -0.2527 0.2886  0.9579
optx  0.4138  0.3888  1.0000 -0.3798 0.4407  0.3862
vac   0.0665 -0.2527 -0.3798  1.0000 0.0806 -0.2360
sqft  0.5353  0.2886  0.4407  0.0806 1.0000  0.2855
x    -0.2354  0.9579  0.3862 -0.2360 0.2855  1.0000
```

```
plot(commpropx)
```



```
lm_out <- lm(rent ~ age + optx + vac + sqft, data = commpropx)
confint(lm_out)
```

```
                  2.5 %      97.5 %
(Intercept) 11.04948640 13.35168536
age         -0.18454113 -0.09952615
optx         0.15619789  0.40783517
vac         -1.54523184  2.78391885
sqft         0.05166283  0.10682321
```

```
lmx_out <- lm(rent ~ age + optx + vac + sqft + x, data = commpropx)
confint(lmx_out)
```

```
                  2.5 %      97.5 %
(Intercept) 11.01749984 13.34724095
age         -0.25790412  0.01159938
optx         0.15623979  0.40983379
vac         -1.54831726  2.81106547
sqft         0.05149706  0.10700398
x           -0.29444125  0.21862943
```

The width of the CI for β_{age} *was* 0.085.

With the new covariate the width of the CI for β_{age} becomes 0.27.

So including `x` in the model makes our estimation of β_{age} less accurate.

```r
summary(lm_out)
```

```
Call:
lm(formula = rent ~ age + optx + vac + sqft, data = commpropx)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.1872 -0.5911 -0.0910  0.5579  2.9441 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 12.20059    0.57796  21.110  < 2e-16 ***
age         -0.14203    0.02134  -6.655 3.89e-09 ***
optx         0.28202    0.06317   4.464 2.75e-05 ***
vac          0.61934    1.08681   0.570     0.57    
sqft         0.07924    0.01385   5.722 1.98e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.137 on 76 degrees of freedom
Multiple R-squared:  0.5847,    Adjusted R-squared:  0.5629 
F-statistic: 26.76 on 4 and 76 DF,  p-value: 7.272e-14
```

The p-value for age is very small.

```r
summary(lmx_out)
```

```

Call:
lm(formula = rent ~ age + optx + vac + sqft + x, data = commpropx)

Residuals:
     Min       1Q   Median       3Q      Max 
-3.10692 -0.60213 -0.04274  0.54459  2.97885 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 12.18237    0.58474  20.834  < 2e-16 ***
age         -0.12315    0.06764  -1.821   0.0727 .  
optx         0.28304    0.06365   4.447 2.97e-05 ***
vac          0.63137    1.09417   0.577   0.5656    
sqft         0.07925    0.01393   5.688 2.34e-07 ***
x           -0.03791    0.12878  -0.294   0.7693    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.144 on 75 degrees of freedom
Multiple R-squared:  0.5852,	Adjusted R-squared:  0.5576 
F-statistic: 21.16 on 5 and 75 DF,  p-value: 3.886e-13
```

The p-value for age is not nearly as small when `x` is included!

Getting VIFs with vif() from the car package

We can use the R package car from Fox and Weisberg (2019).

First time must install the package with install.package("car").

```
library(car)
vif(lm_out)
```

$\frac{1}{1-R_j^2}$

```
     age     optx      vac     sqft 
1.240348 1.648225 1.323552 1.412722 
```

```
vif(lmx_out)
```

```
      age      optx       vac      sqft         x 
12.309581  1.653127  1.325402  1.412727 12.186875 
```

Note the change in VIF for age due to including x in the model!

Variable selection:

$$Y_i = \beta_0 + \underbrace{\beta_1 x_{i1} + \cdots + \beta_p x_{ip}}_{p \text{ covariates}} + \varepsilon_i$$

What if p is very large?

Large p leads to higher VIFs.

Might be good to get rid of some covariates!

Variable selection

Deciding which covariates to keep in the model.

Sometimes the number of potentially important predictors is quite large.

Large p tends to increase the VIFs, leading to low power.

So we may wish to discard some predictors. We briefly discuss:

1. Best subset selection with Mallow's $C(p)$
2. Forward and backward stepwise selection with AIC
3. LASSO selection

And most importantly:

- The dangers of naïve post-selection inference!!

1. Best subset selection : Find best subset of covariates according to some criterion.

$$Y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

Best subset selection with Mallow's C_p

optional

Given q available covariates, there are 2^q possible subset models (why?).

Mallow's C_p can be used to compare subset models: Let

$$C_p = (n-(p+1))\left[\frac{\mathrm{MS}_{\mathrm{Error}}(\mathrm{subset})}{\mathrm{MS}_{\mathrm{Error}}(\mathrm{all})} - 1\right] + (p+1),$$

where

- p is the number of predictors *in the subset model*.
- $\mathrm{MS}_{\mathrm{Error}}(\mathrm{subset})$ is the $\mathrm{MS}_{\mathrm{Error}}$ of the subset model.
- $\mathrm{MS}_{\mathrm{Error}}(\mathrm{all})$ is the $\mathrm{MS}_{\mathrm{Error}}$ of the model with all the covariates.

If the subset model is adequate, $\mathrm{MS}_{\mathrm{Error}}(\mathrm{subset})$ estimates the same target as $\mathrm{MS}_{\mathrm{Error}}(\mathrm{all})$, so the first term should be small and $C_p \approx p+1$.

Can look at C_p values for all subset models of each size $p = 0, 1, 2, \ldots, q$

Want smallest model such that $C_p \approx p+1$.

Mallow's C_p on the rental properties data

optional

Compute Mallow's C_p for a single subset model:

```
lm_all <- lm(rent ~ vac + age + optx + sqft, data = commprop)
lm_sub <- lm(rent ~ age + sqft, data = commprop)
MSE_sub <- sum(lm_sub$residuals^2) / (n - 3)
MSE_all <- sum(lm_all$residuals^2) / (n - 5)
Csub <- (MSE_sub / MSE_all - 1)*(n - 3) + 3
Csub
```

```
[1] 22.87781
```

This value is too large; the subset is not a good one.

The regsubsets() function from the R package leaps

```r
library(leaps) # first time run install.packages("leaps")
regsubsets_out <- regsubsets(rent ~ vac + age + optx + sqft, data = commprop)
summary(regsubsets_out)
```

```
Subset selection object
Call: regsubsets.formula(rent ~ vac + age + optx + sqft, data = commprop)
4 Variables  (and intercept)
     Forced in Forced out
vac      FALSE      FALSE
age      FALSE      FALSE
optx     FALSE      FALSE
sqft     FALSE      FALSE
1 subsets of each size up to 4
Selection Algorithm: exhaustive
         vac age optx sqft   p+1
1  ( 1 ) " " " " " "  "*"     2
2  ( 1 ) " " "*" " "  "*"     3
3  ( 1 ) " " "*" "*"  "*"     4
4  ( 1 ) "*" "*" "*"  "*"     5
```

Want smallest model such that $C_p \approx p + 1$.

```r
summary(regsubsets_out)$cp
```

```
[1] 53.585208 22.877809  3.324753  5.000000
```

FIFA data

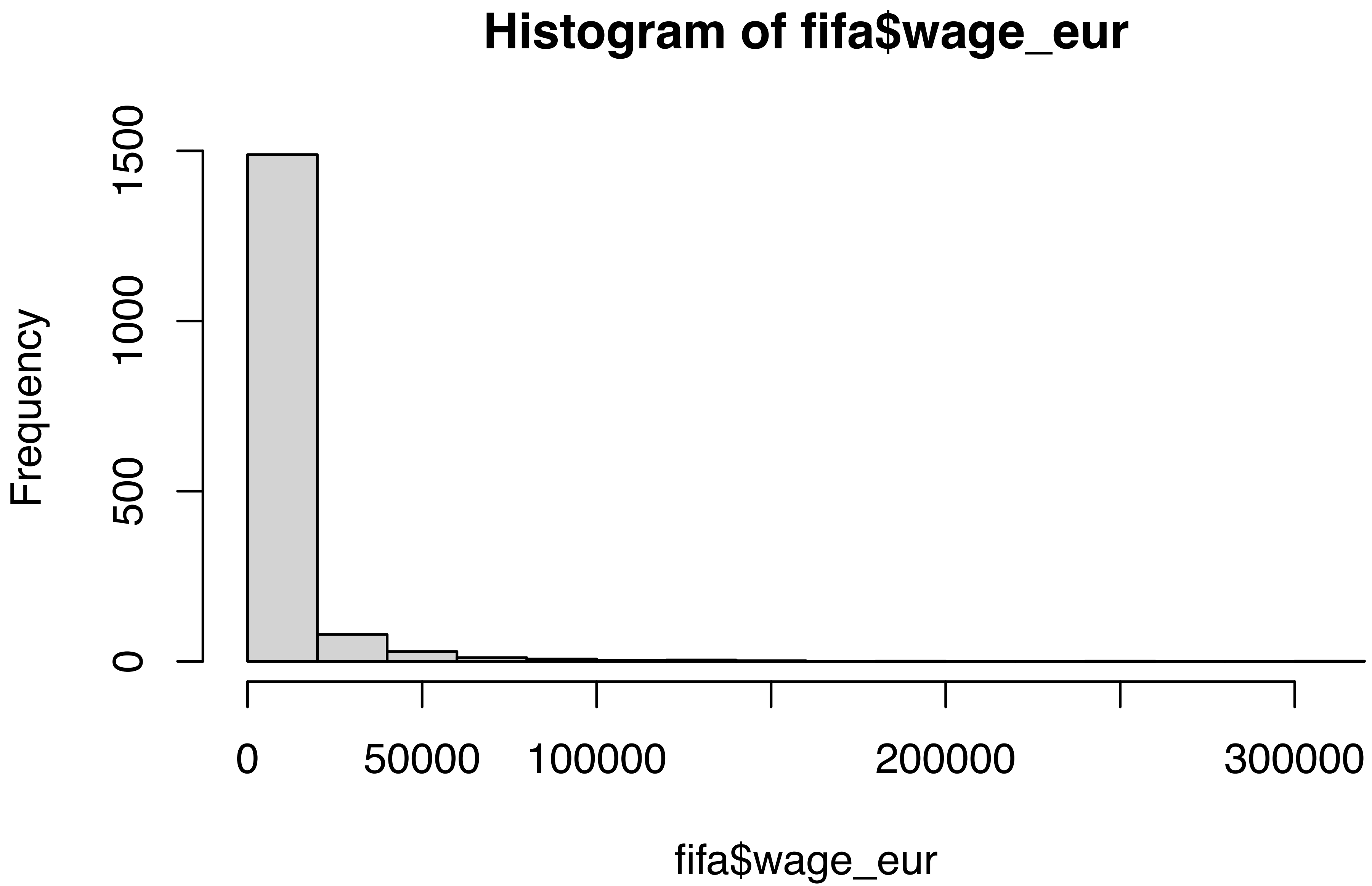
Wages and stats of male FIFA players in 2022 from Pedersen (2022).

```r
link <- url("https://people.stat.sc.edu/gregorkb/data/fifa_usge.csv")
fifa <- read.csv(link)
colnames(fifa)
```

```
 [1] "wage_eur"                  "age"
 [3] "height_cm"                 "weight_kg"
 [5] "nationality_name"          "overall"
 [7] "potential"                 "attacking_crossing"
 [9] "attacking_finishing"       "attacking_heading_accuracy"
[11] "attacking_short_passing"   "attacking_volleys"
[13] "skill_dribbling"           "skill_curve"
[15] "skill_fk_accuracy"         "skill_long_passing"
[17] "skill_ball_control"        "movement_acceleration"
[19] "movement_sprint_speed"     "movement_agility"
[21] "movement_reactions"        "movement_balance"
[23] "defending_standing_tackle" "defending_sliding_tackle"
[25] "goalkeeping_diving"        "goalkeeping_handling"
[27] "goalkeeping_kicking"       "goalkeeping_positioning"
[29] "goalkeeping_reflexes"
```

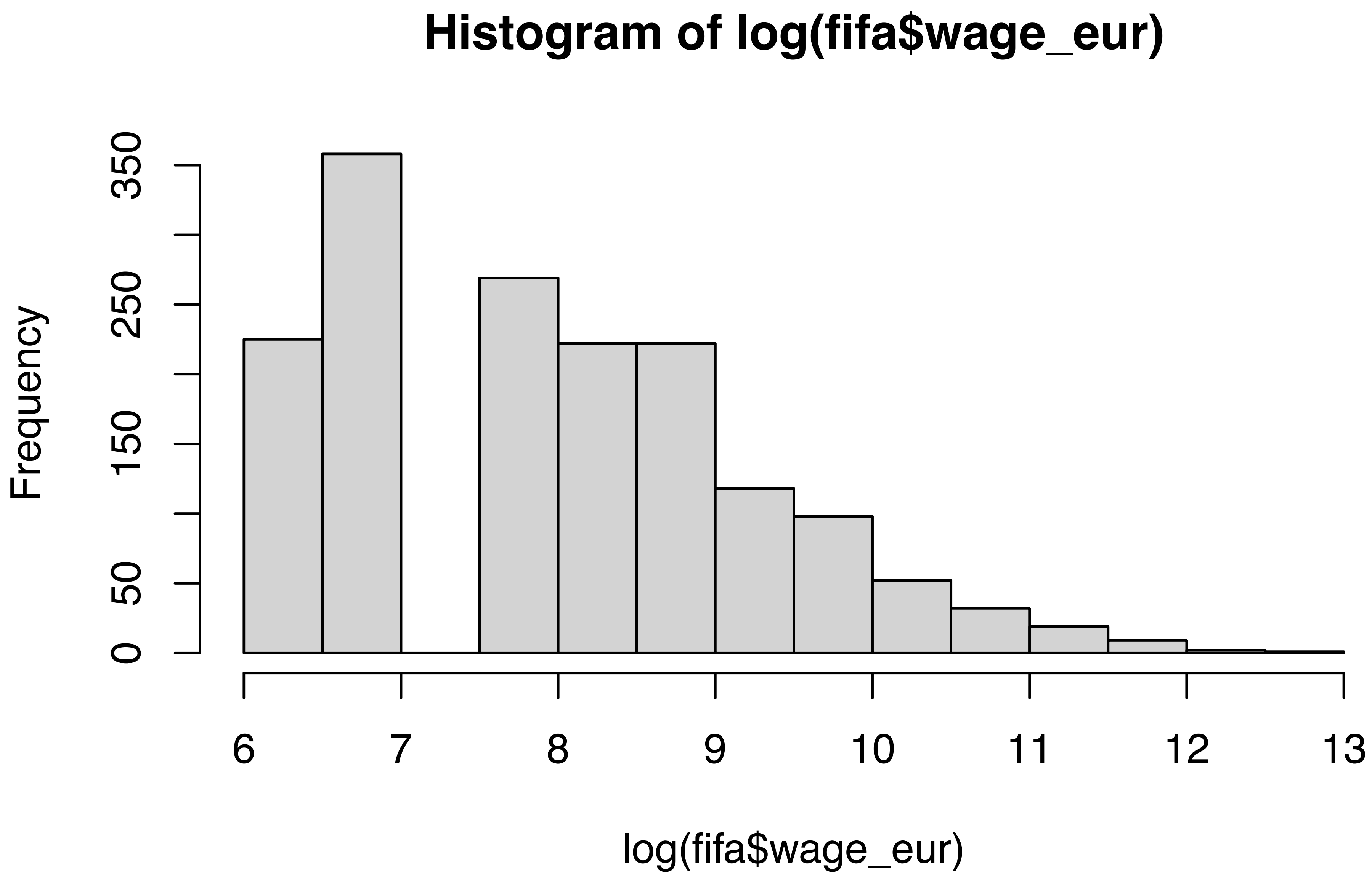
Predict wage from 28 covariates? Too many sub-models to consider!

```
hist(fifa$wage_eur)
```



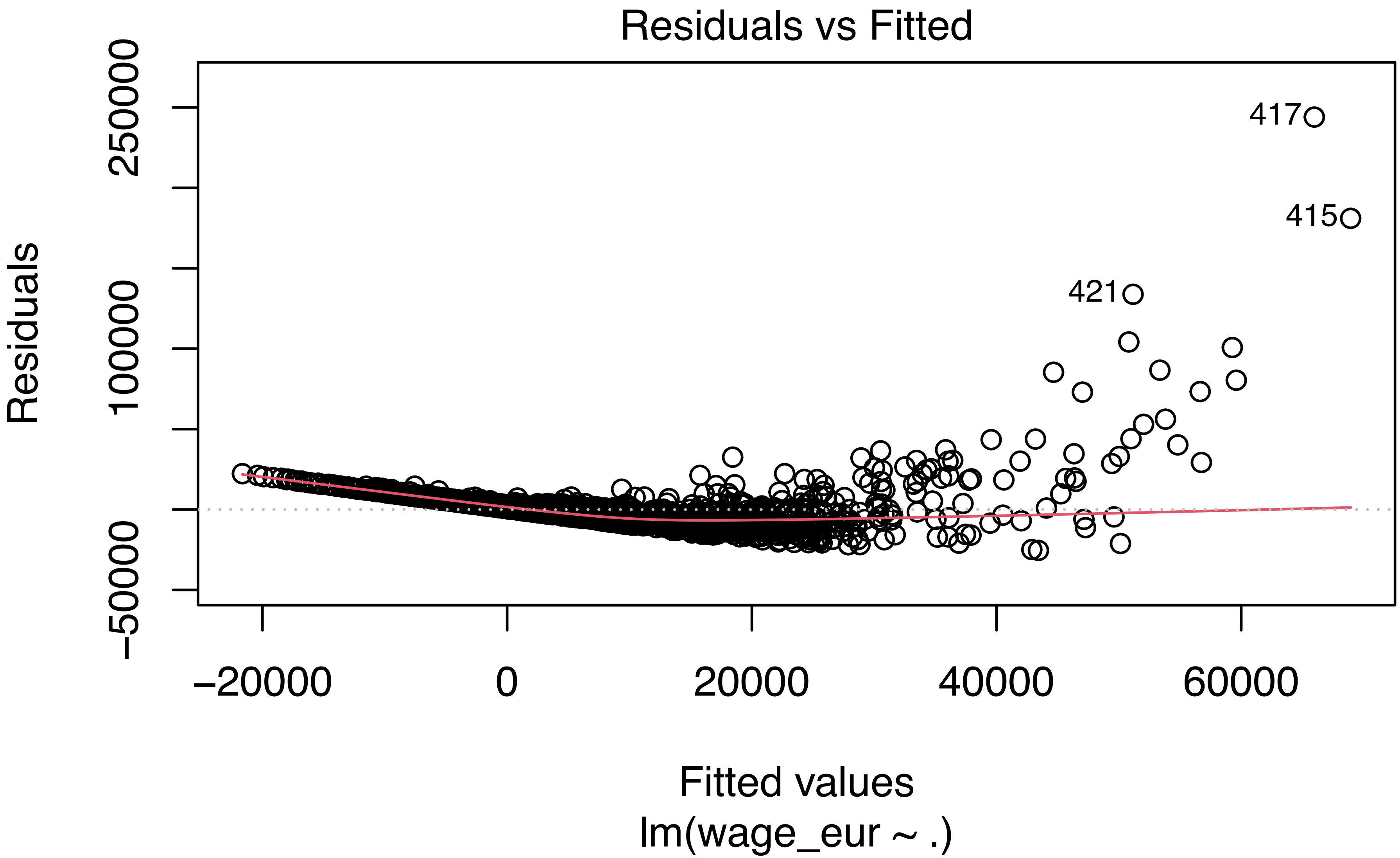
The wage distribution has some high outlying observations.

```r
hist(log(fifa$wage_eur))
```

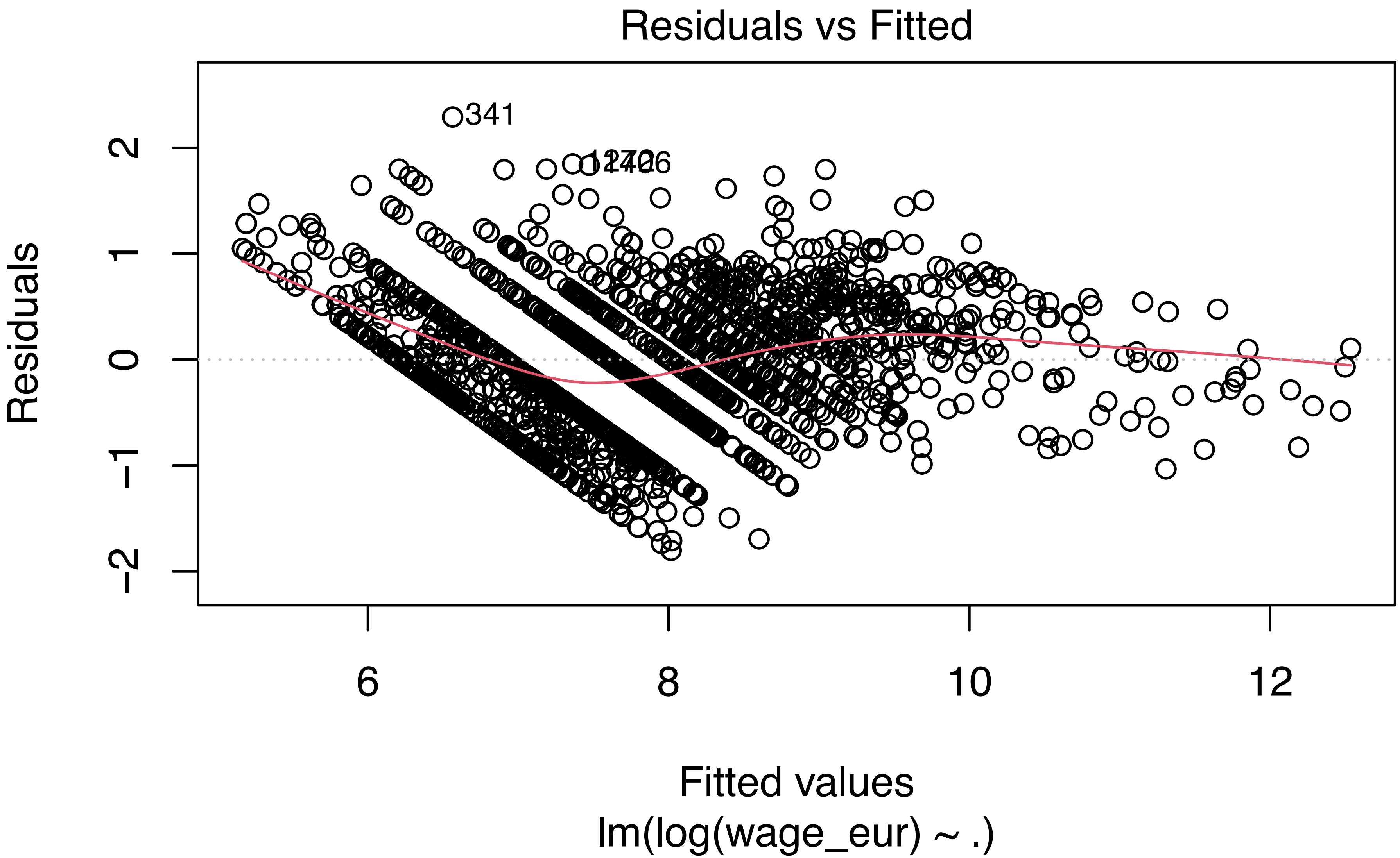


Perhaps better to consider the log of the wage.

```
lm_wage <- lm(wage_eur ~ ., data = fifa)
plot(lm_wage,which = 1)
```



```
lm_logwage <- lm(log(wage_eur) ~ ., data = fifa)
plot(lm_logwage,which = 1)
```



Note the values in the `nationality_name` column:

```r
table(fifa$nationality_name)
```

```
germany     usa
   1214     413
```

R will automatically make an indicator/dummy variable defined as

$$\texttt{nationality\_nameusa}_i = \begin{cases} 1 & \text{if } \texttt{usa} \\ 0 & \text{if } \texttt{german} \end{cases} \qquad \text{for } i = 1, \ldots, n.$$

赤池弘次 (あかいけひろつぐ)

A ka i ke Hi ro tsu gu

Introduced Akaike's Information Criterion (AIC).

Akaike's Information Criterion (AIC) for comparing models

covariates in model

For a given model, i.e. set of covariates, AIC is defined as

STAT 512

$$\text{AIC} = 2(p+1) - 2\,\underbrace{\ell(\hat{\sigma}^2, \hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p)}_{\text{log-likelihood}}\,.$$

The log-likelihood is the log of the joint pdf of the data (STAT 512).

AIC can be used to compare several models for the same data.

The "best" model is the one which minimizes AIC.

Fifa data. # models

1 predictor 2 predictors 3 predictors

$$28 + \binom{28}{2} + \binom{28}{3} + \ldots \binom{28}{28} = 2^{28}$$

The extractAIC() function

The extractAIC function in R returns a modified version of AIC:

$$\text{AIC}^* = 2(p+1) + n\log(\text{SS}_{\text{Error}}/n)$$

```
lm_out <- lm(log(wage_eur) ~ age + potential, data = fifa)
extractAIC(lm_out) # gives value p + 1 as well as AIC value
```

```
[1]    3.0000 -860.2624
```

```
# compute it "manually"
n <- nrow(fifa)
p <- 2
2*(p+1) + n * log(sum(lm_out$residuals^2)/n)
```

```
[1] -860.2624
```

Comparing models using AIC

Compare two models for the FIFA data with AIC:

```
lm1 <- lm(log(wage_eur) ~ age + potential + height_cm, data = fifa)
extractAIC(lm1)
```

```
[1]    4.0000 -858.5413
```

```
lm2 <- lm(log(wage_eur) ~ height_cm + overall, data = fifa)
extractAIC(lm2)
```

```
[1]     3.000 -1377.386
```

The second model has a smaller value of AIC, so it is better according to this criterion.

Stepwise selection based on AIC

Stepwise selection:

- Backward: Begin with all the predictors and remove one at a time.
- Forward: Begin with no predictors and add one at a time.

In each step remove/add predictor to get largest decrease in AIC.

If a decrease in AIC is not possible, stop.

Stepwise selection with fifa data

Use the `step()` function for backward selection:

```
lm_intercept <- lm(log(wage_eur) ~ 1, data = fifa)
lm_all <- lm(log(wage_eur) ~ ., data = fifa)

# backward selection
step_back <- step(lm_all,
                  direction = "backward",
                  scope = formula(lm_all),
                  trace = 0) # suppress printed output
```

```r
summary(step_back)
```

```
Call:
lm(formula = log(wage_eur) ~ age + height_cm + nationality_name +
    overall + potential + attacking_crossing + attacking_finishing +
    attacking_heading_accuracy + attacking_volleys + skill_dribbling +
    skill_fk_accuracy + skill_ball_control + movement_sprint_speed +
    movement_agility + movement_reactions + movement_balance +
    defending_sliding_tackle, data = fifa)

Residuals:
     Min       1Q   Median       3Q      Max
-1.81770 -0.42583 -0.00742  0.44183  2.26000

Coefficients:
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                -1.285472   1.056816  -1.216  0.22403
age                        -0.018033   0.008311  -2.170  0.03018 *
height_cm                  -0.012306   0.005090  -2.418  0.01574 *
nationality_nameusa        -0.102428   0.038424  -2.666  0.00776 **
overall                     0.153692   0.007624  20.159  < 2e-16 ***
potential                   0.025805   0.006486   3.978 7.25e-05 ***
attacking_crossing          0.004266   0.002123   2.009  0.04472 *
attacking_finishing        -0.003433   0.002427  -1.415  0.15740
attacking_heading_accuracy  0.004529   0.001829   2.476  0.01337 *
attacking_volleys           0.005065   0.002464   2.056  0.03995 *
skill_dribbling             0.006814   0.003818   1.785  0.07450 .
skill_fk_accuracy           0.003723   0.001845   2.018  0.04372 *
skill_ball_control         -0.005958   0.004151  -1.435  0.15141
movement_sprint_speed      -0.002731   0.001744  -1.566  0.11754
movement_agility           -0.008042   0.002766  -2.907  0.00370 **
movement_reactions          0.011193   0.003620   3.092  0.00202 **
movement_balance           -0.005163   0.002830  -1.825  0.06824 .
defending_sliding_tackle   -0.004565   0.001446  -3.156  0.00163 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6215 on 1609 degrees of freedom
Multiple R-squared:  0.7725,    Adjusted R-squared:  0.7701
F-statistic: 321.4 on 17 and 1609 DF,  p-value: < 2.2e-16
```

Use the `step()` function for forward selection:

```
# forward selection
step_forw <- step(lm_intercept,
                  direction = "forward",
                  scope = formula(lm_all),
                  trace = 0) # suppress printed output
```

```
summary(step_forw)
```

```
Call:
lm(formula = log(wage_eur) ~ overall + potential + attacking_volleys +
    movement_agility + skill_fk_accuracy + nationality_name +
    movement_reactions + defending_sliding_tackle + attacking_crossing +
    age, data = fifa)

Residuals:
     Min       1Q   Median       3Q      Max
-1.93782 -0.42630 -0.00072  0.44823  2.29004

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)              -3.7485849  0.3310448 -11.323  < 2e-16 ***
overall                   0.1510192  0.0074671  20.225  < 2e-16 ***
potential                 0.0259892  0.0064344   4.039 5.62e-05 ***
attacking_volleys         0.0042858  0.0016009   2.677  0.00750 **
movement_agility         -0.0102281  0.0016175  -6.324 3.30e-10 ***
skill_fk_accuracy         0.0038679  0.0017634   2.193  0.02841 *
nationality_nameusa      -0.0759454  0.0366090  -2.075  0.03819 *
movement_reactions        0.0113347  0.0036018   3.147  0.00168 **
defending_sliding_tackle -0.0026760  0.0009261  -2.889  0.00391 **
attacking_crossing        0.0042171  0.0018421   2.289  0.02219 *
age                      -0.0163448  0.0080810  -2.023  0.04328 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6231 on 1616 degrees of freedom
Multiple R-squared:  0.7703,    Adjusted R-squared:  0.7689
F-statistic:   542 on 10 and 1616 DF,  p-value: < 2.2e-16
```

Forward and backward stepwise selection may give different models!

LASSO selection

The LASSO estimators $\hat{\beta}_0^L, \hat{\beta}_1^L, \dots, \hat{\beta}_p^L$ are obtained by minimizing

$$Q_\lambda(b_0, b_1, \dots, b_p) = \sum_{i=1}^{n}(Y_i - (b_0 + b_1 x_{i1} + \cdots + b_p x_{ip}))^2 + \overset{\text{penalty}}{\boxed{\lambda \sum_{j=1}^{p} |b_j|,}}$$

where $\lambda > 0$ is a tuning parameter.

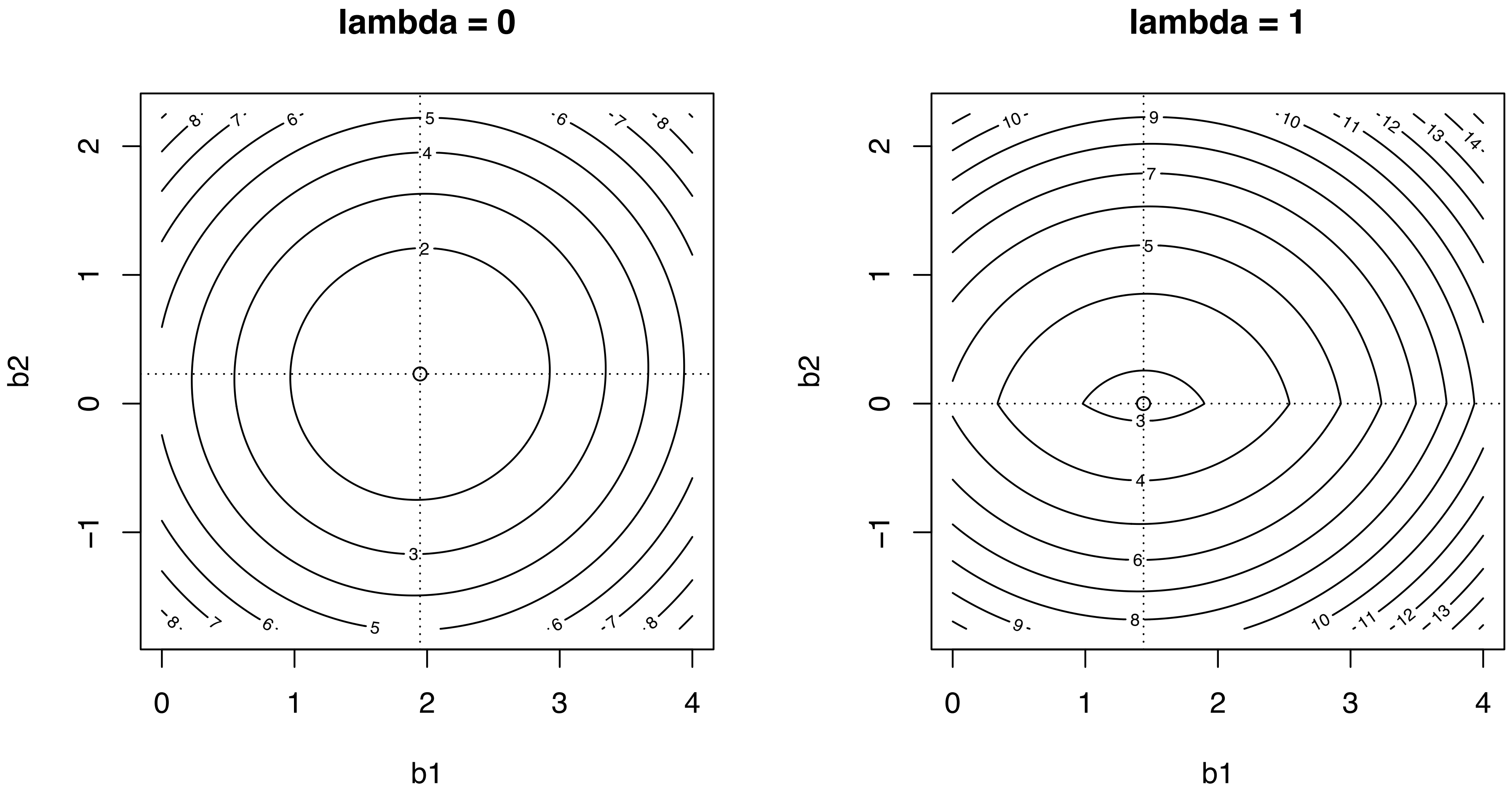
encourages $\hat{\beta}_j = 0$ for some j.

- The penalty term $\lambda \sum_{j=1}^{p} |b_j|$ can cause $\hat{\beta}_j^L = 0$ for some j.
- For λ large enough, all the $\hat{\beta}_j^L$ will be equal to zero.
- So LASSO performs variable selection and estimation simultaneously.
- Drawback: Hard to build CIs based on $\hat{\beta}_0^L, \hat{\beta}_1^L, \dots, \hat{\beta}_p^L$.

Effect of LASSO penalty on the objective function

```
# simulate some data with centered X and centered y (eliminates intercept)
n <- 500;p <- 2
X <- scale(matrix(rnorm(n*p),n,p)); b <- c(2,1/4); e <- rnorm(n)
y <- drop(X %*% b) + e - mean(e)

# define least squares and LASSO objective functions
Q <- function(b,X,y) mean((y - X %*% b)^2)
Qlambda <- function(b,X,y,lambda) Q(b,X,y) + lambda * sum(abs(b))

# set LASSO penalty parameter
lambda <- 1

# evaluate Q and Qlambda over a grid of b1 and b2 values
b1seq <- seq(b[1]-2,b[1]+2,length=200)
b2seq <- seq(b[2]-2,b[2]+2,length=200)
Q_vals <- Qlambda_vals <- matrix(0,length(b1seq),length(b2seq))
for(i in 1:length(b1seq))
  for(j in 1:length(b2seq)){

    Q_vals[i,j] <- Q(b=c(b1seq[i],b2seq[j]),X,y)
    Qlambda_vals[i,j] <- Qlambda(b=c(b1seq[i],b2seq[j]),X,y,lambda)

  }
```

```r
# compute least squares and lasso estimator
bhat <- coef(lm(y~X-1))
bhat_lambda <- optim(par = c(0,0),fn = Qlambda,X = X, y = y,lambda = lambda)$par

# make contour plots of least-squares and LASSO objective functions
par(mfrow=c(1,2))
contour(z = Q_vals, x = b1seq, y = b2seq, main = "lambda = 0",xlab = "b1", ylab = "b2")
points(x = bhat[1],y = bhat[2]);abline(v = bhat[1], lty = 3);abline(h = bhat[2], lty = 3)

contour(z = Qlambda_vals, x = b1seq, y = b2seq, main = paste( "lambda =",lambda), xlab = "b1", ylab = "b2")
points(x = bhat_lambda[1],y = bhat_lambda[2]); abline(v = bhat_lambda[1],lty = 3);abline(h = bhat_lambda[2],lty = 3)
```



LASSO on the FIFA data

Use `cv.ncvreg()` function from R package `ncvreg`.

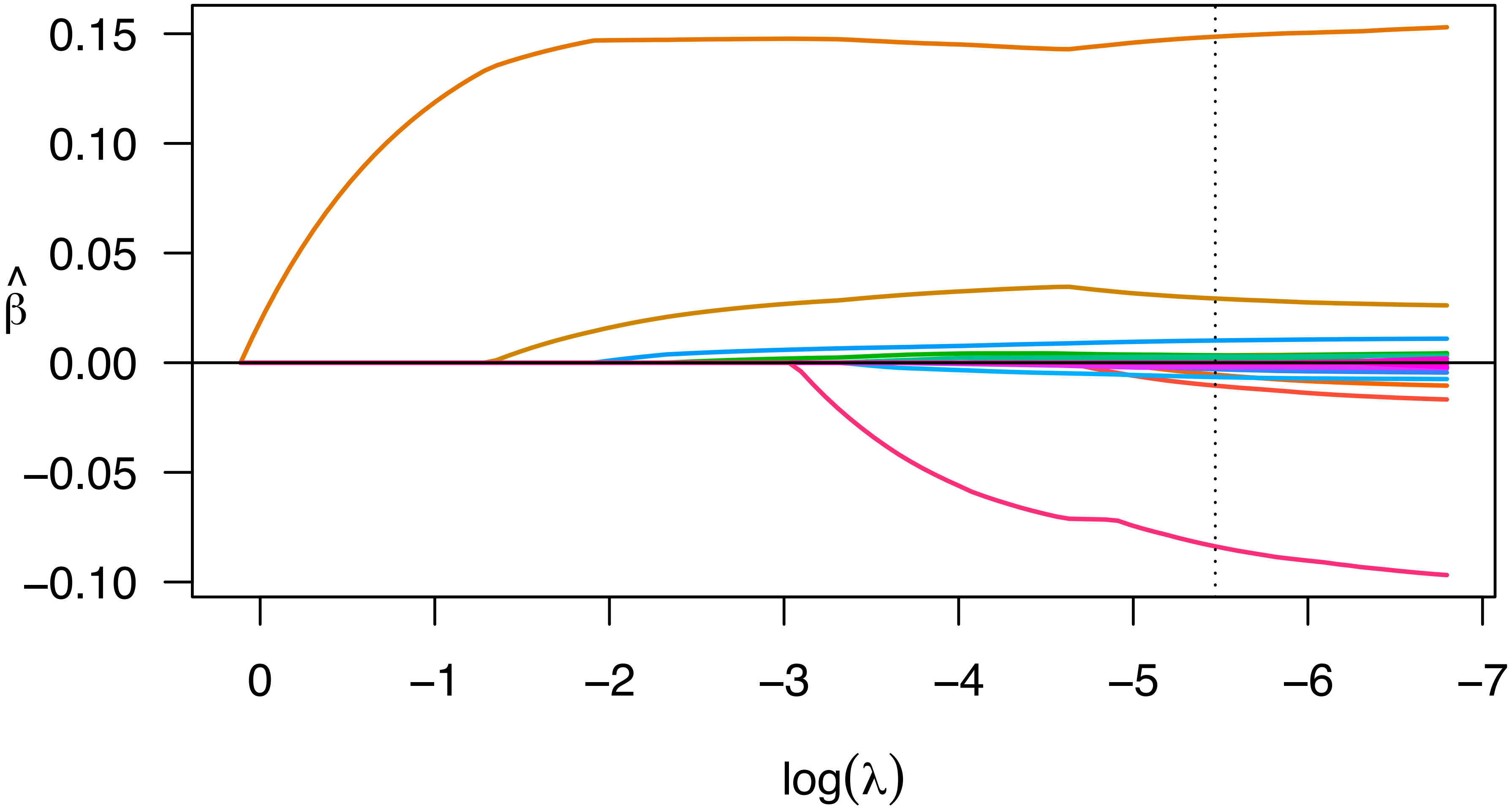
Runs crossvalidation to choose the best value of λ.

```
library(ncvreg) # first time run install.packages("ncvreg")

# prepare response vector and design matrix
y <- log(fifa$wage_eur)
X <- fifa[,-c(1,5)]
X$nationality <- ifelse(fifa$nationality_name == "usa",1,0)

# crossvalidation to choose lambda
lasso <- cv.ncvreg(X,y,penalty = "lasso")
```

```
lasso$fit$beta[,lasso$min] # estimates under the "best" lambda
```

```
               (Intercept)                        age
             -2.8209409392              -0.0104673603
                 height_cm                  weight_kg
             -0.0054578578               0.0000000000
                   overall                  potential
              0.1486544991               0.0292681339
        attacking_crossing        attacking_finishing
              0.0033387453               0.0000000000
attacking_heading_accuracy    attacking_short_passing
              0.0018813248               0.0000000000
         attacking_volleys            skill_dribbling
              0.0033542083               0.0004243427
               skill_curve          skill_fk_accuracy
              0.0008408621               0.0027543693
        skill_long_passing         skill_ball_control
              0.0000000000               0.0000000000
     movement_acceleration      movement_sprint_speed
             -0.0010019224              -0.0011935706
          movement_agility         movement_reactions
             -0.0065415902               0.0101236059
          movement_balance  defending_standing_tackle
             -0.0029825671              -0.0007755575
  defending_sliding_tackle         goalkeeping_diving
             -0.0021010923               0.0000000000
      goalkeeping_handling        goalkeeping_kicking
              0.0000000000               0.0000000000
   goalkeeping_positioning       goalkeeping_reflexes
              0.0000000000               0.0000000000
               nationality
             -0.0837490312
```

```r
plot(lasso$fit,log.l = TRUE)
abline(v = log(lasso$fit$lambda[lasso$min]), lty = 3)
```



The dangers of post-selection inference

It is dangerous to:

1. Ask the data what hypotheses to test (what model to build).
2. Use afterwards the same data to perform inference (get p values).

Illustration:

Add 50 spurious predictors to the commercial properties data.

See how many we find to be significant.

```r
n <- nrow(commprop)
X <- matrix(rnorm(n*50),n,50)
colnames(X) <- paste("x",1:50,sep="")
commpropX <- cbind(commprop,X)

lmX_out <- lm(rent ~ ., data = commpropX)
```

```
summary(lmX_out)
```

```

Call:
lm(formula = rent ~ ., data = commpropX)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.59062 -0.28456  0.05265  0.37467  1.32721 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 12.240240   0.804196  15.220 1.83e-14 ***
age         -0.159799   0.032199  -4.963 3.71e-05 ***
optx         0.306053   0.088151   3.472 0.001821 ** 
vac         -1.019504   1.626735  -0.627 0.536309    
sqft         0.075606   0.019462   3.885 0.000631 ***
x1           0.081065   0.202076   0.401 0.691580    
x2          -0.013859   0.215785  -0.064 0.949282    
x3           0.353769   0.156937   2.254 0.032835 *  
x4          -0.246326   0.214493  -1.148 0.261255    
x5           0.094743   0.217846   0.435 0.667219    
x6          -0.255471   0.179398  -1.424 0.166325    
x7           0.044972   0.249455   0.180 0.858330    
x8          -0.093089   0.173642  -0.536 0.596451    
x9          -0.173610   0.200781  -0.865 0.395125    
x10         -0.456824   0.171506  -2.664 0.013095 *  
x11         -0.198532   0.182312  -1.089 0.286157    
x12          0.167599   0.225407   0.744 0.463821    
x13         -0.042958   0.158114  -0.272 0.788006    
x14         -0.149825   0.157784  -0.950 0.351082    
x15          0.044798   0.206143   0.217 0.829660    
x16         -0.085366   0.179704  -0.475 0.638728    
x17          0.409642   0.198006   2.069 0.048639 *  
x18         -0.014995   0.168287  -0.089 0.929681    
x19          0.310235   0.233058   1.331 0.194696    
x20         -0.095293   0.177272  -0.538 0.595460    
x21         -0.241792   0.201412  -1.200 0.240774    
x22         -0.187829   0.178686  -1.051 0.302854    
x23         -0.057653   0.150114  -0.384 0.704054    
x24          0.015410   0.160248   0.096 0.924129    
x25          0.045967   0.212807   0.216 0.830669    
x26         -0.163131   0.206006  -0.792 0.435601    
x27          0.298277   0.166657   1.790 0.085146 .  
```

We reject H_0: $\beta_j = 0$ at $\alpha = 0.05$ for 3 of the spurious predictors.

So the Type I error rate was $3/50 = 0.06$.

Now do backwards stepwise selection to throw some variables away.

Then see how many of the spurious predictors we find “significant”.

```r
stepX_out <- step(lmX_out, data = commpropX, trace = 0)
summary(stepX_out)
```

```

Call:
lm(formula = rent ~ age + optx + vac + sqft + x3 + x4 + x6 +
    x9 + x10 + x11 + x14 + x17 + x19 + x21 + x22 + x27 + x30 +
    x33 + x35 + x38 + x39 + x40 + x43 + x45, data = commpropX)

Residuals:
     Min       1Q   Median       3Q      Max
-1.56212 -0.38254  0.00396  0.45158  1.45098

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.30833    0.45134  27.271  < 2e-16 ***
age         -0.14311    0.01677  -8.535 1.03e-11 ***
optx         0.28698    0.04891   5.868 2.49e-07 ***
vac         -1.20706    0.88517  -1.364 0.178134
sqft         0.07511    0.01105   6.800 7.41e-09 ***
x3           0.34486    0.10129   3.405 0.001231 **
x4          -0.15437    0.10145  -1.522 0.133739
x6          -0.19038    0.09232  -2.062 0.043839 *
x9          -0.22542    0.11113  -2.029 0.047268 *
x10         -0.38397    0.09581  -4.008 0.000183 ***
x11         -0.29839    0.10528  -2.834 0.006377 **
x14         -0.17005    0.09444  -1.801 0.077133 .
x17          0.41548    0.10696   3.885 0.000273 ***
x19          0.32996    0.11166   2.955 0.004567 **
x21         -0.15718    0.10393  -1.512 0.136084
x22         -0.15554    0.10417  -1.493 0.141020
x27          0.26847    0.08775   3.060 0.003398 **
x30         -0.31647    0.10519  -3.009 0.003927 **
x33         -0.15086    0.09853  -1.531 0.131348
x35         -0.20031    0.10194  -1.965 0.054391 .
x38          0.16086    0.10969   1.466 0.148108
x39          0.12000    0.09388   1.278 0.206457
x40          0.20019    0.10926   1.832 0.072230 .
x43          0.16494    0.10597   1.557 0.125213
x45          0.20619    0.08264   2.495 0.015574 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7825 on 56 degrees of freedom
```

Backwards stepwise selection keeps 20 of the 50 spurios predictors.

Among these 20, we reject H_0: $\beta_j = 0$ at $\alpha = 0.05$ for 10 of them.

So the post-selection Type I error rate was $10/20 = 0.5$ 😱.

WARNING: Selecting variables and then getting p-values in the selected model often leads to astonishingly inflated Type I error rates.

References

Fox, John, and Sanford Weisberg. 2019. *An R Companion to Applied Regression*. Third. Thousand Oaks CA: Sage. https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Kutner, Michael H, Christopher J Nachtsheim, John Neter, and William Li. 2005. *Applied Linear Statistical Models*. McGraw-hill.

Pedersen, Ulrik Thyge. 2022. "FIFA Players." kaggle. https://www.kaggle.com/datasets/ulrikthygepedersen/fifa-players.