Regression via Ordinary Least Squares (OLS)

Gustavo de los Campos

Consider a linear model of the form

$y_i = \mu + x_{i1}\beta_1 + x_{i2}\beta_2 + ... + x_{ip}\beta_p + \varepsilon_i,$

where $i = 1, ..., n$ indexes subjects in the sample.

In matrix form, we can write the above equation as follows

$y_i = \mathbf{x}_i'\beta + \varepsilon_i,$

where $\mathbf{x}_i = (1, x_{i1}, x_{i2}, ..., x_{ip})'$ and $\beta = (\mu, \beta_1, \beta_2, ..., \beta_p)'$.

Stacking the n-data-equations into a system we get

$\mathbf{y} = \mathbf{X}\beta + \varepsilon$

where $\mathbf{y} = (y_1, y_2, ..., y_n)'$ is a $n \times 1$ "response" vector, $\mathbf{X} = [\mathbf{x}_1', \mathbf{x}_2', ..., \mathbf{x}_n']'$ is a $n \times (p+1)$ incidence matrix for the vector of effects $\beta = (\mu, \beta_1, ..., \beta_p)'$.

Example

The Gout data set contains data on serum urate, gout, sex, ethnicity, and age.

```
 DATA=read.table('https://raw.githubusercontent.com/gdlc/STAT_COMP/master/goutData.txt',
                 header=TRUE,sep='')
head(DATA)
```

```
##   sex race age  su gout
## 1   M    W  67 8.3    N
## 2   F    W  72 8.6    N
## 3   F    W  70 7.3    N
## 4   F    W  63 6.2    N
## 5   F    W  55 4.3    N
## 6   M    W  63 7.0    N
```

```
table(DATA$sex)
```

```
##
##   F   M
## 225 175
```

```
table(DATA$race)
```

```
##
##   B   W
##  92 308
```

The incidence matrix of effects

Consider a model for serum urate (`su`) as a function of sex, race and age (`su~sex+race+age`). The variables `sex` and `race`, are categorical, we introduce these variables in the model using dummy variables (as many as the number of levels of the factor minus one). The `model.matrix()` function in R creates incidence matrices for effects from a formula. This is illustrated in the following example.

```
X=model.matrix(~sex+race+age,data=DATA)
y=DATA$su

head(DATA)
```

```
##   sex race age  su gout
## 1   M    W  67 8.3    N
## 2   F    W  72 8.6    N
## 3   F    W  70 7.3    N
## 4   F    W  63 6.2    N
## 5   F    W  55 4.3    N
## 6   M    W  63 7.0    N
```

```
head(X)
```

```
##   (Intercept) sexM raceW age
## 1           1    1     1  67
## 2           1    0     1  72
## 3           1    0     1  70
## 4           1    0     1  63
## 5           1    0     1  55
## 6           1    1     1  63
```

```
head(y)
```

```
## [1] 8.3 8.6 7.3 6.2 4.3 7.0
```

```
dim(DATA)
```

```
## [1] 400   5
```

```
dim(X)
```

```
## [1] 400   4
```

```
length(y)
```

```
## [1] 400
```

Ordinary least squares

Ordinary Least Squares (OLS) estimates are obtained by minimizing the Residual Sum of Squares (RSS),

$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta) = \mathbf{y}'\mathbf{y} + \beta'\mathbf{X}'\mathbf{X}\beta - 2\beta'\mathbf{X}'\mathbf{y})$.

Differentiating the RSS with respect to β leads to

$2\mathbf{X}'\mathbf{X}\beta - \mathbf{2X}'\mathbf{y}$,

setting this equal to zero, leads to the following first-order conditions (FOCs, aka "normal equations"):

$\mathbf{X}'\mathbf{X}\hat{\beta} = \mathbf{X}'\mathbf{y}$

Thus, when $\mathbf{X}$ is a full-column-rank matrix, that is if $\mathbf{X}'\mathbf{X}^{-1}$ exist,

$\hat{\beta} = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y}$

Example 1: using lm() to obtain OLS estimates

```
fm=lm(su~sex+race+age,data=DATA)

coef(fm)
```

```
## (Intercept)        sexM       raceW         age
##  4.31975213  1.52852797 -0.78211876  0.02673734
```

Note: `model.matrix()` chooses the first level of each factor as the reference group, if you wish to choose a different reference, you can change the order of the levels. This is illustrated in the following example:

```
 x=factor(c('a','a','b','c','c'))
 levels(x)
```

```
## [1] "a" "b" "c"
```

```
 model.matrix(~x)
```

```
##   (Intercept) xb xc
## 1           1  0  0
## 2           1  0  0
## 3           1  1  0
## 4           1  0  1
## 5           1  0  1
## attr(,"assign")
## [1] 0 1 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

```
 # now let's use c as refernece
 levels(x)=c('c','b','a')

 model.matrix(~x)
```

```
##   (Intercept) xb xa
## 1           1  0  0
## 2           1  0  0
## 3           1  1  0
## 4           1  0  1
## 5           1  0  1
## attr(,"assign")
## [1] 0 1 1
## attr(,"contrasts")
## attr(,"contrasts")$x
## [1] "contr.treatment"
```

Interpretation of regression coefficients

If a predictor is quantiative (e.g., age, weight), then the corresponding regression coefficient is interpreted as an slope, that is the expected rate of change in y, per unit change in x.

For categorcial predictors, when we use dummy coding (i.e., one dummy variable per level, without including one for the reference level) the corresponding coefficients are interpreted as mean differences. For example, if sex_i is a dummy variable for male ($sex_i = 1$ for male, 0 for female) and we have a linear model of the form: $y_i = \mu + sex_i\beta_1 + age_i\beta_2 + \varepsilon_i$, then β_1 is interpreted as the "male minus female" difference in the expected value of y_i, holding age constant.

Inference

The previous code shows how to obtain a point-estimate (an OLS estimate) from a sample. Our goal is to make inference about the population parameters (that is the regression coefficients in the population from which the sample was drawn). Frequentist inference studies the distribution of estimates over conceptual repeated sampling.

Sampling distribution: We can think of estimators (e.g., least-squares) as a function that maps from data (e.g., $\mathbf{y}$ and $\mathbf{X}$) onto a point-estimate (e.g., $\hat{\beta}$). For every possible sample from the population, the function returns a point estimate. The sampling distribution of an estimator describes how the estimator is expected to vary over conceptual repeated sampling. Important features of this distribution include the expected value and the variance of the estimator.

The expected value of the OLS estimator is

$E[\hat{\beta}|\mathbf{X}] = E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{E}[\mathbf{y}|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{E}[\mathbf{X}\beta + \varepsilon|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}\beta + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'E[\varepsilon|\mathbf{X}]$

Therefore, if either $E[\varepsilon|\mathbf{X}] = 0$ or $\mathbf{X}'E[\varepsilon|\mathbf{X}] = 0$, then

$E[\hat{\beta}] = \beta$,

thus, implying that OLS estimates are unbiased.

The sampling (co)variance matrix of the OLS estimator is:

$Var[\hat{\beta}|\mathbf{X}] = Var[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'Var[\mathbf{y}|\mathbf{X}]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$

Assuming that the error terms are independent, and have homogeneous variance, $Var[\mathbf{y}|\mathbf{X}] = \mathbf{I}_n\sigma_\varepsilon^2$, then

$Var[\hat{\beta}|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\sigma_\varepsilon^2$

Thus, to obtain an estimate of the (co)variance matrix of the estimator we need an estimate of the error variance

An unbiased estimate of the error variance: The expected value of the RSS is

$E[(\mathbf{y} - \mathbf{X}\hat{\beta})'(\mathbf{y} - \mathbf{X}\hat{\beta})|\mathbf{X}] = \sigma_\varepsilon^2 \times (n - rank[\mathbf{X}])$, in the full-rank case $rank[\mathbf{X}] = p + 1$;

therefore, with the notation used here, a method-of moment, unbiased estimator of the error variance is:

$\hat{\sigma}_\varepsilon^2 = RSS/(n - p - 1])$.

Large-sample distribution

Clearly, the estimator, $\hat{\beta} = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y}$, is a weighted sum of the data (y).

According to the Central Limit Theorem the asymptotic distribution of the estimator is Multivariate Normal, the mean is the true parameter (β) because the estimator is unbiased (see previous results), and the (co)variance matrix is

$Var[\hat{\beta}] = (\mathbf{X}'\mathbf{X})^{-1}\sigma_\varepsilon^2$;

therefore,

$\hat{\beta}$~MVN(β,$(\mathbf{X}'\mathbf{X})^{-1}\sigma_\varepsilon^2$)

Note: The CLT does not require any assumptions about the distribution of the error terms (not even independence or homskedasticity are required!). However, if the error terms are IID Normal, then the above result also holds in small samples.

Standard Errors: The SE of each of the coefficients is simply the square root of the corresponding diagonal value of the sampling (co)variance matrix, that is

$SE(\hat{\beta}_j) = \sigma_\varepsilon^2(\mathbf{X}'\mathbf{X})_{jj}^{-1}$

z-statistic: According to the CLT,

$\hat{\beta}_j \sim N(\beta_j, \sigma^2_\varepsilon(\mathbf{X}'\mathbf{X})^{-1}_{jj})$;

therefore,

$z = \hat{\beta}_j / SE(\hat{\beta}_j) \sim N(\beta_j, 1)$.

1DF Tests: A p-value for the (two-sided) test $H_0 : \beta_j = 0$ Vs $H_a : \beta_j \neq 0$ can be obtained from the standard normal distribution

```
pValue=pnorm(q=abs(z),lower.tail=FALSE)*2
```

Now, to compute the SE we replace σ^2_ε with an estimate ($\hat{\sigma}^2_\varepsilon$, see above); therefore, instead of using the standard normal distribution, we should use the t-distribution with degrees of freedom equal to the residual degree of freedom (n-p-1), thus

```
pValue=pt(q=abs(z),df=nrow(X)-ncol(X),lower.tail=FALSE)*2
```

For large sample size (e.g., n-p-1>30) the t converges to the standard normal distribution and the difference in the p-value computed from one or the other would be small.

Example 2: using lm() to obtain OLS estimates, SE, and p-values

Fitting the model and examining estimates, SEs, and p-values

```
fm=lm(su~sex+race+age,data=DATA)
class(fm)
```

```
## [1] "lm"
```

```
summary(fm)
```

```
##
## Call:
## lm(formula = su ~ sex + race + age, data = DATA)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4843 -0.9717 -0.1829  0.8276  5.4296
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.31975    0.81533   5.298 1.95e-07 ***
## sexM         1.52853    0.14306  10.684  < 2e-16 ***
## raceW       -0.78212    0.16932  -4.619 5.22e-06 ***
## age          0.02674    0.01299   2.058   0.0402 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.413 on 396 degrees of freedom
## Multiple R-squared:  0.2504, Adjusted R-squared:  0.2447
## F-statistic: 44.09 on 3 and 396 DF,  p-value: < 2.2e-16
```

Residuals and predictions

```
# Predictions and residuals for the training data
eHat=residuals(fm) # try help(residuals) for options about different type of residuals, more on this b
yHat=predict(fm)
```

```r
# You can also derive predicitons for new data
tmp=data.frame(age=c(50,55),sex=c('F','F'),race=c('W','B'))
predict(fm,newdata=tmp)
```

```
##        1        2
## 4.874501 5.790306
```

Diagnostic plots

For residual diagnostics we can call `plot()` on the `lm` objecct.

```r
plot(fm)
```

Retrieving the variance co-variance matrix of estimates

```r
vcov(fm)
```

```
##               (Intercept)          sexM         raceW           age
## (Intercept)  0.664764024 -0.0038349056 -0.0081685585 -0.0103951575
## sexM        -0.003834906  0.0204667219 -0.0020227609 -0.0000568041
## raceW       -0.008168559 -0.0020227609  0.0286690772 -0.0002076739
## age         -0.010395158 -0.0000568041 -0.0002076739  0.0001687320
```

```r
## Correlation of estimates
cov2cor(vcov(fm))
```

```
##              (Intercept)        sexM       raceW         age
## (Intercept)  1.00000000 -0.03287735 -0.05917044 -0.98151874
## sexM        -0.03287735  1.00000000 -0.08350521 -0.03056728
## raceW       -0.05917044 -0.08350521  1.00000000 -0.09442268
## age         -0.98151874 -0.03056728 -0.09442268  1.00000000
```

The Hat matrix

Predictions in a linear model take the form

$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$,

replacing $\hat{\beta}$ with the OLS estimate, we get

$\hat{\mathbf{y}} = \mathbf{X}[\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y} = \mathbf{H}\mathbf{y}$

where,

$\mathbf{H} = \mathbf{X}[\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'$ is the 'Hat matrix'.

This Hat matrix is symmetric (i.e., $\mathbf{H} = \mathbf{H}'$), positive semi-definite (i.e., $\alpha'\mathbf{H}\alpha \geq 0$) and idempotent (implying that $\mathbf{HH} = \mathbf{H}$).

Using $\mathbf{H}$, model residuals residuals can be represented as follows:

$\hat{\varepsilon} = (\mathbf{y} - \hat{\mathbf{y}}) = \mathbf{y} - \mathbf{H}\mathbf{y} = [\mathbf{I}_n - \mathbf{H}]\mathbf{y}$.

Studentized residuals

The (co)varinance matrix of model residuals is $Cov(\hat{\varepsilon}) = [\mathbf{I}_n - \mathbf{H}]Cov(\mathbf{y})[\mathbf{I}_n - \mathbf{H}]'$,

assuming independent and homoscedasticity residuals, we get,

$Cov(\hat{\varepsilon}) = \sigma^2_\varepsilon[\mathbf{I}_n - \mathbf{H}]\mathbf{I}_n[\mathbf{I}_n - \mathbf{H}]' = \sigma^2_\varepsilon[\mathbf{I}_n - \mathbf{H}]$;

therefore, the variance of the ith predicted residual is

$Var(\hat{\varepsilon}_i) = \sigma^2_\varepsilon(1 - H_{ii})$.

Studentized residuals are standarized to unit variance; thus they are defined as:

$\tilde{\varepsilon}_i = \hat{\varepsilon}_i / \sqrt{\sigma^2_\varepsilon(1 - H_{ii})}$.

```
eStd=rstudent(fm)
head(eStd)
```

Hypothesis testing in linear models

Hypotheses involving a single coefficient (i.e., 1-DF tests) can be tested using the t-test (or normal test in case of large samples) discussed above. However, many times we want to test hypotheses involving more than one DF. For example, consider a model for an outcome (`y`) as a fuction of two predictors, a factor with 4 levels, and a quantitative predictor. We can represent this model as follows

$H_a : y_i = \mu + x_{1i}\beta_1 + x_{2i}\beta_2 + x_{3i}\beta_3 + x_{4i}\beta_4 + \varepsilon_i$

where x_{1i}, x_{2i}, and x_{3i} are dummy variables associated to three levels of the factor, and x_{3i} is the quantitative covariate. To test whether the factor in question has any effect on the outcome we need to test H_a against

H_0 $y_i = \mu + x_{3i}\beta_3 + \varepsilon_i$,

Or, simply stated, $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$. Likelihood ratio test, F-test, and Wald's tests can be used to test this hypothesis. Here, we discuss F test.

F-test

The F-test uses the ratio of two scaled sum of squares: the sum of squares of the model (scaled by the model degrees of freedom) and the residual sum of squares (scaled by the residual degree of freedom).

To implement an F-test, we fit the model under the null and alternative hypotheses, and obtain, from each of these models the corresponding residual sum of squares (say RSS_0 from H_0 and RSS_a from H_a). The amount of variance explained by the factor, after accunting for the effects in H_0 is: $MSS = RSS_0 - RSS_a$ (the 'model sum of squares'), the residual sum of squares is simply the RSS under H_a (i.e., RSS_a).

The degree of freedom of RSS_a equals $df_2 = n - p - 1$, where p is the number of predictors in H_a, 4 in the example above. On the other hand, $(RSS_0 - RSS_a)$ has $df_1 = p_a - p_0$ (i.e., the difference in the number of parameters between H_a and H_0 ($df_1 = 3$), 3 in the example considered above.

Under the null hypothesis both $(RSS_0 - RSS_a)/\sigma^2_\varepsilon$ and $RSS_a/\sigma^2_\varepsilon$ are independent, and, under H_0, those statistics follow chi-square distributions with df equal to df_1 and df_2, respectively; terefore, the ratio $[(RSS_0 - RSS_a)/(p_a - p_0)]/[RSS_a/(n - p_a - 1)]$ follows an F-distribution, with $df1 = p_a - p_0$ and $df2 = n - p_a - 1$.

Example

For this example, we simulate data with sample size n, a factor (say, ethnicity) with 4 levels, and a covariate (say, age).

```
n=100
ethnicity=sample(c('Black','White-European','Hispanic','Asian'),size=n,replace=TRUE)
age=rgamma(n=100,shape=50,scale=.5)
X=model.matrix(~ethnicity+age)
b=c('mu'=100,'Black'=0,'Hispanic'=1,'White'=0,'age'=1)
signal=X%*%b
error=rnorm(n,sd=sd(signal)) #simulates a model R-sq of 0.5
y=signal+error
HA=lm(y~ethnicity+age)
H0=lm(y~age)
```

```
RSS0=sum(residuals(H0)^2)
RSSA=sum(residuals(HA)^2)
MSS=RSS0-RSSA
df1=length(coef(HA))-length(coef(H0))
df2=n-length(coef(HA))
Fstat=(MSS/(df1))/(RSSA/(df2))
pValue=pf(q=Fstat,lower.tail=FALSE,df1=df1,df2=df2)

print(c('F'=Fstat,'df1'=df1,'df2'=df2,'pvalue'=pValue))
```

```
##          F        df1        df2     pvalue
##  0.1038442  3.0000000 95.0000000  0.9576316
```

The anova() function

The `anova(fm1,fm2)` takes two lm objects, and compares them using an F-test.

```
print(anova(H0,HA))
```

```
## Analysis of Variance Table
##
## Model 1: y ~ age
## Model 2: y ~ ethnicity + age
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     98 752.73
## 2     95 750.27  3    2.4604 0.1038 0.9576
```