Lecture 3-e
OLS – MLE & Data Problems

Brooks (4th edition): Chapters 3 & 4

© R. Susmel, 2020 (for private use, not to be posted/shared online).[1]

Review: OLS – Summary

• OLS $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}$ (kx1) vector

• Properties for $\mathbf{b}$.

1) Unbiased: $E[\mathbf{b}|\mathbf{X}] = \boldsymbol{\beta}$

2) Efficiency (& BLUE): $Var[\mathbf{b}|\mathbf{X}] = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$

3) If (**A5**) $\boldsymbol{\varepsilon}|\mathbf{X} \sim i.i.d.\ N(\mathbf{0}, \sigma^2\mathbf{I}_T)$ $\Rightarrow \mathbf{b}|\mathbf{X} \sim i.i.d.\ N(\boldsymbol{\beta}, \sigma^2(\mathbf{X}'\mathbf{X})^{-1})$

4) Consistent: $\mathbf{b} \xrightarrow{p} \boldsymbol{\beta}$

5) Asymptotic Normality: $\mathbf{b} \xrightarrow{a} N(\boldsymbol{\beta}, \sigma^2(\mathbf{X}'\mathbf{X})^{-1})$

• Testing H_0 about $\mathbf{b}$, with a *t-test*. For example, $H_0: \beta_k = \beta_k^0$, $H_1: \beta_k \neq \beta_k^0$

$$t_k = \frac{b_k - \beta_k^0}{\text{Est. SE}[b_k]} | \mathbf{X} \sim t_{T-k}$$

Review: Goodness of Fit of the Regression

• After estimating the model (**A1**), we would like to judge the adequacy of the model. There are two ways to do this:

- Visual: Plots of fitted values and residuals, histograms of residuals.
- Numerical measures: R^2, Adjusted R^2, AIC, BIC, etc.

• Numerical measures. In general, they are simple and easy to compute. We call them *goodness-of-fit* measures. Most popular: R^2.

• Definition: Variation

In the context of a model, we consider the *variation* of a variable as the movement of the variable, usually associated with movement of another variable.

Review: Goodness of Fit of the Regression

• Total variation = Total sum of squares (TSS) = $\sum_i (y_i - \bar{y})^2$.

We want to decompose TSS in two parts: one explained by the regression and one unexplained by the regression.

• TSS $= \sum_i (y_i - \bar{y})^2 = \sum_i (y_i - \hat{y}_i + \hat{y}_i - \bar{y})^2$
$= \sum_i (y_i - \hat{y}_i)^2 + \sum_i (\hat{y}_i - \bar{y})^2 + 2\sum_i (y_i - \hat{y}_i)(\hat{y}_i - \bar{y})$
$= \sum_i e_i^2 + \sum_i (\hat{y}_i - \bar{y})^2$

since $\sum_i (y_i - \hat{y}_i)(\hat{y}_i - \bar{y}) = \sum_i e_i\,(\hat{y}_i - \bar{y}) = 0$

Or **TSS** = RSS + SSR

RSS: Residual Sum of Squares (also called SSE: SS of errors)

SSR: Regression Sum of Squares (also called ESS: *explained* SS)

Review: R^2

- **TSS** = SSR + RSS

• We want to have a measure that describes the fit of a regression. Simplest measure: the standard error of the regression (SER)

$$SER = \sqrt{\frac{RSS}{T-k}} = \sqrt{s^2} \qquad \Rightarrow \text{SER depends on units. Not good!}$$

• R-squared (R^2)

1 = SSR/**TSS** + RSS/**TSS**

R^2 = SSR/**TSS** = Regression variation/Total variation

R^2 = 1 – RSS/**TSS**

As introduced here, R^2 lies between 0 and 1 (& it is independent of units of measurement!). It measures how much of total variation (**TSS**) is explained by regression (SSR): the higher R^2, the better.

Review: R^2 – Sensitivity to Adding Regressors

- $R^2 = \frac{SSR}{\mathbf{TSS}}$

Interpretation: The percentage of total variation (TSS) of $\boldsymbol{y}$ explained by the variation of regressors.

Note: R^2 is bounded by zero and one only if:

(a) There is a constant term in **X**.

(b) The line is computed by OLS.

• Main problem with R^2: Adding regressors

It can be shown that R^2 never falls when regressors (say **z**) are added to the regression. This occurs because RSS decreases with more "information" (in the sense of more regressors).

Problem: Judging a model based on R^2 tends to over-fitting.

Review: R^2 – Remarks

• Comparing Regressions

- Make sure the denominator in R^2 is the same - i.e., same left hand side variable. For example, when modeling sales, it is common to use log(Sales). Cannot compare R^2 to the one with Sales. Loglinear will almost always appear to fit better, taking logs reduces variation.

• Linear Transformation of data does not change R^2.

- Get same R^2 with $\mathbf{X}$ or with $\mathbf{X}^* = c\mathbf{X}$.

• Interpretation: The percentage of total variation (TSS) of $\boldsymbol{y}$ explained by the variation of regressors.

• Main problem with R^2: Adding regressors

Adjusted R-squared

• R^2 is modified with a penalty for number of parameters: *Adjusted-R^2*

$$\overline{R}^2 = 1 - \frac{(T-1)}{(T-k)}(1-R^2) = 1 - \frac{(T-1)}{(T-k)}\frac{RSS}{TSS} = 1 - \frac{s^2}{TSS/(T-1)}$$

$\Rightarrow$ maximizing $\overline{R}^2$ <=> minimizing $[RSS/(T-k)] = s^2$

• $\overline{R}^2$ includes a penalty for variables that do not add much fit. Can fall when a variable is added to the equation.

• It will rise when a variable, say $\mathbf{z}$, is added to the regression if and only if the *t-ratio* on $\mathbf{z}$ is larger than one in absolute value.

Adjusted R-squared

• Theil (1957) shows that, under certain assumptions (an important one: the true model is being considered), if we consider several linear models:

M_1: $\boldsymbol{y} = \mathbf{X}_1\boldsymbol{\beta}_1 + \varepsilon_1$ - true model

M_2: $\boldsymbol{y} = \mathbf{X}_2\boldsymbol{\beta}_2 + \varepsilon_2$

M_3: $\boldsymbol{y} = \mathbf{X}_3\boldsymbol{\beta}_3 + \varepsilon_3$

& choose the model with smaller s^2 (or, larger Adjusted R^2), we select the true model, M_1, on average.

• In this sense, we say that "maximizing Adjusted R^2" is an *unbiased* model-selection criterion.

Other Goodness of Fit Measures

• There are other goodness-of-fit measures that also incorporate penalties for number of parameters (degrees of freedom). We minimize these measures.

• Popular Information Criteria (IC)

- **Akaike Information Criterion (AIC)**

$AIC = -2/T(\ln L - k)$ *L: Likelihood*

$\Rightarrow$ if normality $AIC = \ln(\boldsymbol{e'e}/T) + (2/T)\,k$ (+constants)

- **Bayes-Schwarz Information Criterion (BIC)**

$BIC = -(2/T \ln L - [\ln(T)/T]\,k)$

$\Rightarrow$ if normality $AIC = \ln(\boldsymbol{e'e}/T) + [\ln(T)/T]\,k$ (+constants)

Goodness of Fit Measures – Example

Example: 3 Factor F-F Model (continuation) for IBM returns:

```
b <- solve(t(x)%*% x)%*% t(x)%*%y          # b = (X'X)^-1 X' y  (OLS regression)
e <- y - x%*%b                             # regression residuals, e
k <- ncol(x)                               # Number of parameters estimated
RSS <- as.numeric(t(e)%*%e)                # RSS
R2 <- 1 - as.numeric(RSS)/as.numeric(t(y)%*%y)   # R-squared w/ TSS approximation
Adj_R2 <- 1 - (T-1)/(T-k)*(1-R2)           # Adjusted R-squared
AIC <- log(RSS/T) + 2*k/T                  # AIC under N(.,.) –i.e., under (A5)
```

```
> R2
[1] 0.338985
> Adj_R2
[1] 0.3354752
> AIC
[1] -5.671036
```

⇒ The 3 F-F factors explain **34%** of the variability of IBM returns.

Maximum Likelihood Estimation (MLE)

• Idea: Assume a particular distribution with unknown parameters. Maximum likelihood (ML) estimation chooses the set of parameters that maximize the likelihood of drawing a particular sample.

Example: Suppose we have a sample with N realizations of a coin flip. The coin was flipped N=100 times: 60 heads (H) & 40 tails (T).

We know the distribution of the RV X = number of H & T from N coin flips. It follows a binomial distribution, with parameter p, the probability of a head.

$$P[X = x, N] = \binom{N}{x} p^x (1-p)^{N-x}$$

MLE estimates p as the probability that maximizes what we observed in our particular sample. In our case, MLE set p = 0.60.

Maximum Likelihood Estimation (MLE)

Example (continuation):

$$P[X = x, N|p] = \binom{N}{x} p^x (1-p)^{N-x}$$

To check our intuition that $p = 0.60$, we compute $P[X = 60, N = 100]$ for different p:

$p = 0.50$ $\quad P[X = 60, N = 100\,] = \binom{100}{60}.50^{60}\,(.50)^{40} = 0.010844$

$p = 0.55$ $\quad P[X = 60, N = 100\,] = \binom{100}{60}.55^{60}\,(.45)^{40} = 0.048803$

$p = 0.60$ $\quad P[X = 60, N = 100\,] = \binom{100}{60}.60^{60}\,(.40)^{40} =$ **0.081219**

$p = 0.65$ $\quad P[X = 60, N = 100\,] = \binom{100}{60}.65^{60}\,(.35)^{40} = 0.047392$

$p = 0.70$ $\quad P[X = 60, N = 100\,] = \binom{100}{60}.70^{60}\,(.30)^{40} = 0.008491$

It checks! Instead of using trial & error, it will be easier to find p that maximize $P[X = x, N|p]$ using standard calculus ($\Rightarrow \hat{p}_{MLE} = k/N$).

MLE: Maximizing Likelihood Function

• Formally speaking, we create a function that describes the likelihood of observing the sample results. In the coin flip example: $X = x$:

$$L(X = x, N|p) = \binom{N}{x} p^x\,(1-p)^{N-x}$$

Then, we maximize $L(X = x, N|p)$ with respect to p.

• More general, let's consider a sample $(X_1, X_2, \ldots, X_N)$ which is drawn from a distribution (pdf) $f(\boldsymbol{X}|\theta)$, where θ are k unknown parameters. Then, each X_i's has a pdf $f(X_i|\theta)$.

If the X_i's are *independent* with $f(X_i|\theta)$, the joint pdf for the whole sample $(X_1, X_2, \ldots, X_N)$ is:

$$L(\boldsymbol{X}|\theta) = f(X_1, X_2, \ldots, X_N|\theta) = f(X_1|\theta) * f(X_2|\theta) * \cdots * f(X_N|\theta)$$
$$= \prod_{i=1}^{N} f(X_i|\theta)$$

MLE: Maximizing Likelihood Function

• Assuming the X_i's are *independent* with $f(X_i|\theta)$, the joint pdf is:

$$L(\boldsymbol{X}|\theta) = f(X_1|\theta) * f(X_2|\theta) * \cdots * f(X_N|\theta) = \prod_{i=1}^{N} f(X_i|\theta)$$

The function $L(\boldsymbol{X}|\theta)$ is called the *likelihood function*. It represents how likely it is to get a particular sample from the model. This function $L(\boldsymbol{X}|\theta)$ is maximized with respect to θ to obtain ML estimates: $\hat{\theta}_{MLE}$.

It is often easier to work with the *Log of the likelihood* function. That is,

$$\ln L(\boldsymbol{X}|\theta) = \sum_{i=1}^{N} \ln f(X_i|\theta)$$

Then, we maximize as usual:

1st-derivative $\Rightarrow \frac{\partial \ln L(X|\theta)}{\partial \theta} = \sum_{i=1}^{N} \frac{\partial \ln f(X_i|\theta)}{\partial \theta}$

f.o.c. $\Rightarrow \frac{\partial \ln L(X|\hat{\theta}_{MLE})}{\partial \theta} = 0 \quad \Rightarrow \hat{\theta}_{MLE}$.

MLE: Example I

Let the sample be $\boldsymbol{X} = \{5, 6, 7, 8, 9, 10\}$ drawn from a Normal(μ, 1). The probability of each of these points based on the unknown mean, μ, can be written as:

$$f(5|\mu) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{(5-\mu)^2}{2}\right]$$
$$f(6|\mu) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{(6-\mu)^2}{2}\right]$$
$$\vdots$$
$$f(10|\mu) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{(10-\mu)^2}{2}\right]$$

Assume that the sample is *independent*. Then, the joint pdf is given by:

$$L(X|\mu) = f(5|\mu) * f(6|\mu) * \ldots * f(10|\mu)$$

MLE: Example I

Then, the joint pdf function can be written as:

$$L(X|\mu) = \frac{1}{(2\pi)^{6/2}} \exp\left[-\frac{(5-\mu)^2}{2} - \frac{(6-\mu)^2}{2} - \cdots - \frac{(10-\mu)^2}{2}\right]$$

The value of μ that maximizes the likelihood function of the sample can then be defined by $\max_{\mu} L(X|\mu)$.

It easier to maximize the *Log likelihood*, $\ln L(X|\mu)$:

$$\max_{\mu} \ln\big(L(X|\mu)\big) = -{}^{6}/_{2} \ln(2\pi) + \left[-\frac{(5-\mu)^2}{2} - \frac{(6-\mu)^2}{2} - \cdots - \frac{(10-\mu)^2}{2}\right]$$

1st-derivative $\Rightarrow \frac{\partial}{\partial \mu}\left[K - \frac{(5-\mu)^2}{2} - \frac{(6-\mu)^2}{2} - \cdots - \frac{(10-\mu)^2}{2}\right]$

f.o.c. $\Rightarrow$ $(5-\hat{\mu}_{MLE}) + (6-\hat{\mu}_{MLE}) + \cdots + (10-\hat{\mu}_{MLE}) = 0$

MLE: Example I

Then, the first order conditions:

$$(5-\hat{\mu}_{MLE}) + (6-\hat{\mu}_{MLE}) + \cdots + (10-\hat{\mu}_{MLE}) = 0$$

Solving for $\hat{\mu}_{\text{MLE}}$:

$$\hat{\mu}_{MLE} = \frac{5+6+7+8+9+10}{6} = 7.5 = \bar{x}$$

That is, the MLE estimator $\hat{\mu}_{\text{MLE}}$ is equal to the sample mean. This is good for the sample mean: MLE has very good properties!

Remark: In general, finding the MLE estimator, $\hat{\theta}_{MLE}$, analytically, like we do above for $\hat{\mu}_{MLE}$, is not feasible. We use numerical methods to solve the first order conditions for $\hat{\theta}_{MLE}$.

MLE: Remarks

• ML estimation approach is general. All we need is to assume that the data we have follow a distribution. In our CLM context, we need a model (say, **A1**) and a pdf for the errors (say, normal) to use MLE. MLE picks the betas that maximize the likelihood.

Remark: We like MLE because its estimators, $\hat{\theta}_{MLE}$, have very good properties.

• A lot of applications in finance and economics: Time series, volatility (GARCH and stochastic volatility) models, factor models of the term structure, switching models, option pricing, logistic models (mergers and acquisitions, default, etc.), trading models, etc.

• In general, we rely on numerical optimization to get MLEs.

Ronald A. Fisher, England (1890 – 1962)

MLE: Properties

• ML estimators (MLE) have very appealing properties:

(1) **Efficiency.** Under general conditions, they achieve lowest possible variance for an estimator.

(2) **Consistency.** As the sample size increases, the MLE converges to the population parameter it is estimating:

$$\hat{\theta}_{MLE} \xrightarrow{p} \theta$$

(3) **Asymptotic Normality***:* As the sample size increases, the distribution of the MLE converges to the normal distribution.

$$\hat{\theta}_{MLE} \xrightarrow{a} N(\theta, [N\,\mathbf{I}(\theta|x_i)\,]^{-1}) = N(\theta, \mathbf{I}(\theta|X)^{-1})$$

where $\mathbf{I}(\theta|x_i)$ is the *Information matrix* for observation x_i:

$$E\left[\left(\frac{\partial \log f(\theta|x_i)}{\partial \boldsymbol{\theta}}\right)\left(\frac{\partial \log f(\theta|x_i)}{\partial \boldsymbol{\theta}}\right)^{\mathrm{T}}\right] = \mathbf{I}(\theta|x_i) \quad (k\text{x}k \text{ matrix})$$

MLE: Properties

and $$E\left[\left(\frac{\partial \log L}{\partial \boldsymbol{\theta}}\right)\left(\frac{\partial \log L}{\partial \boldsymbol{\theta}}\right)^{\mathrm{T}}\right] = \mathbf{I}(\theta|X)$$

is the information matrix for the whole sample.

$\Rightarrow$ $\quad \mathrm{SE}[\hat{\theta}_{MLE,k}\,|\,\mathbf{X}] = \mathrm{sqrt}\{\mathrm{diag}([\mathbf{I}(\theta|X)^{-1}]_{kk})\}$

(4) **Invariance.** The ML estimate is invariant under functional transformations. That is, if $\hat{\theta}_{MLE}$ is the MLE of θ and if $g(\theta)$ is a function of θ, then $g(\hat{\theta}_{MLE})$ is the MLE of $g(\theta)$.

Example: Suppose we estimated $\hat{\sigma}^2_{MLE}$ -i.e., the MLE of σ^2. Then, $\hat{\sigma}_{MLE} = \mathrm{sqrt}(\hat{\sigma}^2_{MLE})$

(5) **Sufficiency.** If a single sufficient statistic exists for θ, the MLE of θ must be a function of it. That is, $\hat{\theta}_{MLE}$ depends on the sample observations only through the value of a sufficient statistic.

MLE: Numerical Optimization

• We have a function, $f(X|\boldsymbol{\theta}) = \ln L(X|\boldsymbol{\theta})$, with k *unknown* parameters. We use *numerical optimization* to estimate $\boldsymbol{\theta}$.

Numerical optimization are algorithms that search over the parameter space of $\boldsymbol{\theta}$ looking for the values that maximize/minimize $f(X|\boldsymbol{\theta})$.

• Most common optimization algorithms are based on the **Newton-Raphson method** (N-R). It is an iterative algorithm:

- At iteration $j+1$, based on information from the previous iteration j, N-R updates the estimate of $\boldsymbol{\theta}$.

$\boldsymbol{\theta}_{j+1} = \boldsymbol{\theta}_j$ + update (update a function of 1st & 2nd derivatives

At iteration j=1, we input an initial guess)

- N-R stops when the values of $\boldsymbol{\theta}$ at j is similar to the value at $j-1$.

MLE: Numerical Optimization

• In R, the functions *optim* & *nlm* do numerical optimization. Both **minimize** any non-linear function $f(X|\boldsymbol{\theta})$. Recall that max $f(X|\boldsymbol{\theta})$ = min - $f(X|\boldsymbol{\theta})$. Then, in practice, we numerically minimize the negative of the likelihood function, or $\ln L(X|\boldsymbol{\theta})$ * **(-1)**.

Example: In Example I, we numerically minimize $\ln L(X|\mu)$ * **(-1)**.

• To run *optim* or *nlm*, we need to specify:
- Initial values for the parameters, θ_0.
- Function to be minimized (in Example I, $\ln L(X|\mu)$ * **(-1)**).
- Data used.
- Other optional inputs: Choice of method, Hessian calculated, etc.

• More on this topic in Lecture 10.

23

MLE: Numerical Optimization – Code in R

Example: For $\boldsymbol{X}$ = {5, 6, 7, 8, 9, 10} ~ N(μ, 1), code to get $\hat{\mu}_{MLE}$.

```
mu <- 0                          # assumed mean (initial value, needed input to start minim.)
x_6 <- c(5, 6, 7, 8, 9, 10)      # data
dnorm(5, mu, sd=1)               # probability of observing a 5, assuming a N(mu=0, sd=1)
dnorm(x_6)                       # probability of observing each element in x_6
l_f <- prod(dnorm(x_6))          # Likelihood function
log(l_f)                         # Log likelihood function
sum(log(dnorm(x_6)))             # Alternative calculation of Log likelihood function

# Step 1 - Create Likelihood function
likelihood_n <- function(mu){    # Create a prob function with mu as an argument
sum(log(dnorm(x_6, mu, sd=1)))
}
> likelihood_n(mu)               # print likelihood
[1] -183.0136
```

MLE: Numerical Optimization – Code in R

Example (continuation):

```
negative_likelihood_n <- function(mu){   # R uses a minimization algorithm, change sign
sum(log(dnorm(x_6, mu, sd=1))) * (-1)
}
> negative_likelihood_n(mu)
[1] 183.0136

# Step 2 - Maximize (or Minimize negative Likelihood function)
results_n <- nlm(negative_likelihood_n, mu, stepmax=2)      # nlm minimizes the function
> results_n                                                 # Show nlm results
$minimum
[1] 14.26363                <=  The minimized value of function (-14.26363 is the max)
$estimate
[1] 7.5                     <= The MLE for μ (=μ̂_MLE).
$gradient
[1] -4.736952e-12           <= Should be very close to zero if we're at a minimum
```

MLE: Numerical Optimization – Code in R

Example (continuation):

```
mu_max <- results_n$estimate          # Extract estimates
> mu_max                              # Should be equal to mean
[1] 7.5
> likelihood_n(mu_max)                # Check max value at mu_max
[1] -14.26363
```

MLE: Computing the MLE Variance

• To obtain the variance of $\hat{\theta}_{MLE}$ we invert the information matrix for the whole sample $\mathbf{I}(\theta|X)$. Recall,

$$\hat{\theta}_{MLE} \xrightarrow{a} N(\theta,\ \mathbf{I}(\theta|X)^{-1})$$

where $\mathbf{I}(\theta|X)$ is the *Information matrix* for the whole sample. It is generally calculated as:

$$E\left[-\left(\frac{\partial^2 \ln L(\theta|X)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'}\right)\right] = \mathbf{I}(\theta|X), \qquad (k\text{x}k \text{ matrix})$$

where the matrix of second derivatives is the Hessian matrix, **H**:

$$\frac{\partial^2 \ln L(\theta|X)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} = \mathbf{H}$$

• $\mathbf{I}(\theta)$, the *information matrix* (negative expected value of Hessian), measures the shape of the likelihood function. Its inverse gives the variance of the MLE estimator:

MLE: Computing the MLE Variance

• The inverse gives the variance of the MLE estimator:

$$\text{Var}(\hat{\theta}_{MLE}) = E[-\mathbf{H}]^{-1} = \mathbf{I}(\theta)^{-1}$$

• We use numerical optimization packages (say, *nlm* in R), which minimize a function. Then, we *minimize* the *negative* log $L(\theta|X)$ and, thus, to get Var$[\hat{\theta}_{MLE}]$ we do not need to multiply **H** by **(-1)**.

$$\Rightarrow \qquad \text{SE}[\hat{\theta}_{MLE,k}\,|\,\mathbf{X}] = \text{sqrt}\{\text{diag}([\mathbf{H}^{-1}]_{kk})\}$$

Remark: To compute Var$(\hat{\theta}_{MLE})$ we use the inverse of **H**, evaluated at $\hat{\theta}_{MLE}$, as the estimator of the variance. R calculates the Hessian in all optimization packages (for example, *nlm*). In Example I, to compute Var$(\hat{\mu}_{MLE})$ we extract the Hessian from *nlm* with

```
coeff_hess <- results_n$hessian          # Extract Hessian
```

MLE: Example II - Estimating μ & σ^2 In R

Example: For $\boldsymbol{X}$ = {5, 6, 7, 8, 9, 10} ~ N(μ, σ^2), code to get MLEs.

```
mu <- 0                          # assumed mean (initial value)
sig <- 1                         # assumed sd (initial value)
x_6 <- c(5, 6, 7, 8, 9, 10)

# Step 1 - Create Likelihood function
likelihood_lf <- function(x){    # Create a prob function with mu & sig as arguments
mu <- x[1]
sig <- x[2]
sum(log(dnorm(x_6, mu, sd=sig)))
}

negative_likelihood_lf <- function(x){    # R uses a minimization algorithm, change sign
mu <- x[1]
sig <- x[2]
sum(log(dnorm(x_6, mu, sd=sig))) * (-1)
}
negative_likelihood_lf(x)
```

MLE: Example II - Estimating μ & σ^2 In R

Example (continuation):

```
# Step 2 - Maximize Log Likelihood function (or Minimize negative Likelihood function)
results_lf <- nlm(negative_likelihood_lf, x, stepmax=4)        # nlm minimizes the function
> results_lf                                                   # displays nlm results
$minimum
[1] 11.72496                          <=  Minimized value of function

$estimate
[1] 7.500000 1.707825                 <= MLEs for μ & σ² (=μ̂_MLE & σ̂²_MLE)


$gradient
[1] -1.846772e-07 -7.986103e-08       <= ≈ 0 if we're at a minimum

$code
[1] 1                                 <= = 1 if we program  stopped at a minimum

$iterations
[1] 34                                <= Number of iterations
```

MLE: Example II - Estimating μ & σ^2 In R

Example (continuation):

```
# Step 2 (continuation) - Maximize (or Minimize negative Likelihood function)
par_max <- results_lf$estimate                    # Extract estimates
> par_max                                         # Should be equal to sample mean
[1] 7.500000 1.707825
> likelihood_lf(par_max)                          # Check max value of likelihood function
[1] -11.72496

# Step 3 – Standard Errors (by inverting the Hessian)
results_lf <- nlm(negative_likelihood_lf, x, stepmax=4 , hessian=TRUE)
coeff_hess <- results_lf$hessian                  # Extract Hessian
> coeff_hess                                      # Show Hessian
              [,1]         [,2]
[1,]  2.0571428731 -0.0009030531
[2,] -0.0009030531  4.1122292411
cov_lf <- solve(coeff_hess)                       # invert Hessian to get cov(MLEs)
```

MLE: Example II - Estimating μ & σ^2 In R

Example (continuation):

```
# Step 3 (continuation) – Standard Errors (by inverting the Hessian)
cov_lf <- solve(coeff_hess)                       # Invert hess to get cov(MLE estimates)
> cov_lf                                          # Show covariance
 [,1]         [,2]
[1,] 0.4861111549 0.0001067493
[2,] 0.0001067493 0.2431771208
se_lf <- sqrt(diag(cov_lf))                       # Compute S.E. of MLE estimates
> se_lf
[1] 0.6972167 0.4931299

# t-tests
> par_max[1]/se_lf[1]                             # t-ratio for mu
[1] 10.75706
par_max[2]/se_lf[2]                               # t-ratio for sigma2
[1] 3.463236
```

MLE: Example III – CLM + Normal

• We write the CLM, assuming (A5), using matrix notation:

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim N(0, \sigma^2 \mathbf{I}_T)$$

where we have k explanatory, exogenous variables, $\boldsymbol{x}_i$'s , that we treat as numbers. $\boldsymbol{\beta}$ is a kx1 vector of unknown parameters.

Then, the joint likelihood function becomes:

$$L = \prod_{i=1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\varepsilon_i^2}{2\sigma^2}\right) = (2\pi\sigma^2)^{-T/2} \prod_{i=1}^{T} \exp\left(-\frac{\varepsilon_i^2}{2\sigma^2}\right)$$

• Taking logs, we have the log likelihood function:

$$\ln L = -\frac{T}{2}\ln 2\pi\sigma^2 - \frac{1}{2\sigma^2}\sum_{i=1}^{T}\varepsilon_i^2 = -\frac{T}{2}\ln 2\pi\sigma^2 - \frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{2\sigma^2}$$

$$= -\frac{T}{2}\ln 2\pi\sigma^2 - \frac{\mathbf{y}'\mathbf{y} - 2\,\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta}}{2\sigma^2}$$

MLE: Example III – CLM + Normal

• The joint likelihood function becomes:

$$\ln L = -\frac{T}{2}\ln 2\pi\sigma^2 - \frac{\mathbf{y}'\mathbf{y} - 2\,\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta}}{2\sigma^2}$$

• We take 1st derivatives of the log likelihood w.r.t. β and σ^2:

$$\frac{\partial \ln L}{\partial \beta} = -\frac{1}{2\sigma^2}(-2\,\mathbf{X}'\mathbf{y} + 2\,\mathbf{X}'\mathbf{X}\boldsymbol{\beta}) = \frac{1}{\sigma^2}\mathbf{X}'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

$$\frac{\partial \ln L}{\partial \sigma^2} = -\frac{T}{2\sigma^2} - \left(-\frac{\sum_{i=1}^{T}\varepsilon_i^2}{2\sigma^4}\right) = \left(\frac{1}{2\sigma^2}\right)\left[\frac{\sum_{i=1}^{T}\varepsilon_i^2}{\sigma^2} - T\right]$$

Note: $\frac{\partial \ln L}{\partial \theta}$ is a $(k+1)$x1 vector of first derivatives, where $\theta=(\beta, \sigma^2)$.

We set f.o.c. (set $\frac{\partial \ln L}{\partial \theta} = 0$) and, then, solve for that $\widehat{\boldsymbol{\beta}}_{MLE}$ and $\hat{\sigma}^2_{MLE}$.

MLE: Example III – CLM + Normal

• After some algebra, we get:

$$\widehat{\boldsymbol{\beta}}_{MLE} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{y}$$

$$\hat{\sigma}^2_{MLE} = \frac{\sum_{i=1}^{T} e_i^2}{T} = \frac{\sum_{i=1}^{T}(y_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}}_{MLE})^2}{T}$$

• Under (**A5**) –i.e., normality for the errors–, we have that $\widehat{\boldsymbol{\beta}}_{MLE} = \mathbf{b}$.

• This is a good result for OLS **b**. ML estimators are: Efficient, consistent, asymptotically normal and invariant.

• $\hat{\sigma}^2_{MLE}$ is biased, but given that it is an ML estimator, it is efficient, consistent and asymptotically normally distributed.

• It can be shown (see next slides) that Var[$\widehat{\boldsymbol{\beta}}_{MLE}$] = $\hat{\sigma}^2_{MLE}$ $(\boldsymbol{X}'\boldsymbol{X})^{-1}$

MLE: Example III – Computing the Variance

• To get SE for $\hat{\theta}_{MLE}$, we invert the $(k+1)$x$(k+1)$ information matrix:

$$I(\theta|X) = E[-\frac{\partial \ln L}{\partial\theta\partial\theta'}] = \begin{bmatrix} (\frac{1}{\sigma^2}\boldsymbol{X}'\boldsymbol{X}) & 0 \\ 0 & \frac{T}{2\sigma^4} \end{bmatrix}$$

Technical Note: It is block-diagonal, the inverse is the inverse of the diagonal blocks. Then,

$$\text{Var}[\widehat{\boldsymbol{\beta}}_{MLE}] = \hat{\sigma}^2_{MLE}\ (\boldsymbol{X}'\boldsymbol{X})^{-1}$$

$$I(\theta|X)^{-1} = \begin{bmatrix} \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1} & 0 \\ 0 & \frac{2\sigma^4}{T} \end{bmatrix} \quad \text{Var}[\hat{\sigma}^2_{MLE}] = 2\,\hat{\sigma}^4_{MLE}/\text{T}$$

R Note: to compute Var($\hat{\theta}_{MLE}$) we extract the Hessian from *nlm* with

```
coeff_hess <- lf$hessian          # Extract Hessian from MLE object lf
```

MLE: 3-Factor F-F Model + Normal

Example: We estimate the 3 F-F factor model for IBM.

```
SFX_da <-
read.csv("http://www.bauer.uh.edu/rsusmel/4397/Stocks_FX_1973.csv",head=TRUE,sep=",")
x_ibm <- SFX_da$IBM
x_Mkt_RF<- SFX_da$Mkt_RF
x_SMB <- SFX_da$SMB
x_HML <- SFX_da$HML
x_RF <- SFX_da$RF

T <- length(x_ibm)
lr_ibm <- log(x_pfe[-1]/x_pfe[-T])
x0 <- matrix(1,T-1,1)
 Mkt_RF <- x_Mkt_RF[-1]/100
 SMB <- x_SMB[-1]/100
 HML <- x_HML[-1]/100
 RF <- x_RF[-1]/100
ibm_x <- lr_ibm - RF
X <- cbind(x0, Mkt_RF, SMB, HML)
```

MLE: 3-Factor F-F Model + Normal

Example (continuation):

```
# Step 1 - Negative Likelihood function
likelihood_lf <- function(theta, y ,X) {
N <- nrow(X)
k <- ncol(X)
beta <- theta[1:k]
sigma2 <- theta[k+1]^2
e <- y - X%*%beta
logl <- -.5*N*log(2*pi) - .5*N*log(sigma2) - ((t(e)%*%e)/(2*sigma2))
return(-logl)                                   # Negative log likelihood
}

theta <- c(0,1,1,1,.1)                          # initial values
likelihood_lf(theta, ibm_x, X)
             [,1]
[1,] -599.0825
```

MLE: 3-Factor F-F Model + Normal

Example (continuation):

```
# Step 2 - Maximize (or Minimize negative Likelihood function)
results_lf <- nlm(likelihood_lf, theta, hessian=TRUE, y=ibm_x, X=X)      # nlm minimizes l_f
par_max <- results_lf$estimate                                           # Extract estimates
> par_max                                                                # Should be equal to OLS results
[1] -0.0005907974  0.8676052091 -0.6815947799 -0.2284249895  0.0557422421
> likelihood_lf(par_max,ibm_x,X)                                         # Check max value of likelihood function
 [,1]
[1,] -835.3316
```

MLE: 3-Factor F-F Model + Normal

Example (continuation):

```
# Step 3 - Compute S.E. by inverting Hessian
par_hess <- results_lf$hessian                                           # Extract Hessian
> par_hess                                                               # Show Hessian matrix
              [,1]        [,2]             [,3]                [,4]         [,5]
[1,] 183123.2131 1034.3403801 300.5280632 452.9161743 -3.243494e+02
[2,]   1034.3404  390.1995683  71.3131499 -55.6126338 -6.913297e-01
[3,]    300.5281   71.3131499 170.5839168 -26.9486009 -3.023956e-01
[4,]    452.9162  -55.6126338 -26.9486009 165.2938181 -2.928687e-01
[5,]   -324.3494   -0.6913297  -0.3023956  -0.2928687  3.629895e+05
cov_lf <- solve(par_hess)                                                # invert Hessian to get covariance
se_lf <- sqrt(diag(cov_lf))                                              # Compute standard errors
> se_lf
[1] 0.002370939 0.054063912 0.080170161 0.080713227 0.001659791

> (par_max[2] -1)/se_lf[2]                                               # t-test for H0: beta=1
[1] -2.448857
```

MLE: 3-Factor F-F Model + Normal

Example (continuation):

```
# Compare with OLS results
> fit_ibm_ff3 <- lm(ibm_x ~ Mkt_RF + SMB + HML)
> summary(fit_ibm_ff3)

Coefficients:
             Estimate     Std. Error   t value Pr(>|t|)
(Intercept) -0.0005903   0.0023793   -0.248  0.80416
Mkt_RF       0.8676042   0.0542554   15.991  < 2e-16 ***
SMB         -0.6815950   0.0804542   -8.472  < 2e-16 ***
HML         -0.2284263   0.0809992   -2.820  0.00497 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.05594 on 565 degrees of freedom
Multiple R-squared:  0.3519,    Adjusted R-squared:  0.3485
F-statistic: 102.3 on 3 and 565 DF,  p-value: < 2.2e-16
```

MLE: 3-Factor F-F Model + Normal

Example (continuation):

• Summary: OLS vs MLE

	OLS		MLE	
	Coeff. (1)	S.E.	Coeff. (2)	S.E.
Intercept	-0.00509	0.00238	-0.00509	0.00237
Mkt_RF	0.86761	0.05425	0.86761	0.05406
SMB	-0.68159	0.08045	-0.68159	0.08017
HML	-0.22842	0.08100	-0.22842	0.08071

Same as expected

Note: $\hat{\sigma}^2_{MLE}$ is biased, but as T gets bigger, the differences between $\hat{\sigma}^2_{MLE}$ and s^2_{OLS} become very small. Thus, with a big T (& normality) the difference between Var[$\widehat{\boldsymbol{\beta}}_{MLE}$] & Var[$\boldsymbol{b}$] should be minor.

Data Problems

"*If the data were perfect, collected from well-designed randomized experiments, there would hardly be room for a separate field of econometrics.*" Zvi Griliches (1986, **Handbook of Econometrics**)

- Three important data problems:

(1) **Missing Data** – very common, especially in cross sections and long panels.

(2) **Outliers** - unusually high/low observations.

(3) **Multicollinearity** - there is perfect or high correlation in the explanatory variables.

- In general, data problems are exogenous to the researcher. We cannot change the data or collect more data.

Missing Data

- *General Setup*

We have an indicator variable, s_i:

If $s_i = 1$, we observe Y_i,

If $s_i = 0$, we do not observe Y_i.

Note: We always observe the missing data indicator s_i.

- Suppose we are interested in the population mean $\theta = \mathrm{E}[Y_i]$.

- With a lot of information -large T-, we can learn $p = \mathrm{E}[s_i]$ and $\mu_1 = \mathrm{E}[Y_i \mid s_i = 1]$, but nothing about $\mu_0 = \mathrm{E}[Y_i \mid s_i = 0]$.

- We can write: $\theta = p * \mu_1 + (1 - p) * \mu_0$.

Problem: Even in large samples we learn nothing about μ_0. Without additional information/assumptions we cannot say much about θ.

Missing Data

• Without additional information/assumptions there is no much we can say about θ.

• Now, suppose the variable of interest is binary: $Y_i \in \{0, 1\}$. We also have an explanatory variable of Y_i, say W_i.

• Then, the natural (not data-informed) lower and upper bounds for μ_0 are 0 and 1 respectively. This implies bounds on θ:

$$\theta \in [\theta_{LB}, \theta_{UB}] = [p * \mu_1, \; p * \mu_1 + (1 - p) * 1].$$

• These bounds are *sharp*, in the sense that without additional information we cannot improve on them.

If from variable W_i we can infer something about the missing values, these bounds can be improved.

Missing Data: CLM

• Now, suppose we have the CLM: $y_i = \boldsymbol{x}_i' \boldsymbol{\beta} + \varepsilon_i$

• We use the selection indicator, s_i, where $s_i = 1$ if we can use observation i. After some algebra we get,

$$\mathbf{b} = \boldsymbol{\beta} + (\sum_{i=1}^{T} s_i \, \boldsymbol{x}_i' \boldsymbol{x}_i / T)^{-1} (\sum_{i=1}^{T} s_i \, \boldsymbol{x}_i' \varepsilon_i / T)$$

• For unbiased (and consistent) results, we need $E[s_i \, \boldsymbol{x}_i' \, \varepsilon_i] = 0$, implied by $E[\varepsilon_i | \, s_i \, \boldsymbol{x}_i'] = 0$ (*)

In general, we find that when $s_i = h(\boldsymbol{x}_i)$, that is, the selection is a function of $\boldsymbol{x}_i$, we have an inconsistent OLS $\mathbf{b}$. This situation is called *selection bias*.

Missing Data: CLM

Example of Selection Bias: Determinants of Hedging.

A researcher only observes companies that hedge. Estimating the determinants of hedging from this population will bias the results!

• Q: When it is safe to ignore the problem? If missing observations are randomly (exogenously) "selected." Rubin (1976) calls this assumption "*missing completely at random*" (or MCAR).

In general, MCAR is rare. In general, it is more common to see "*missing at random,"* where missing data depends on observables (say, education, sex) but one item for individual *i* is NA (Not Available).

If in the regression we "control" for the observables that influence missing data (not easy), it is OK to delete the whole observation for *i*.

Missing Data: Usual Solutions

Otherwise, we can:

a. Fill in the blanks –i.e., *impute* values to the missing data- with averages, interpolations, or values derived from a model.

b. Use (inverse) probability weighted estimation. Here, we inflate or "over-weight" unrepresented subjects or observations.

c. Heckman selection correction: Build a model for the selection function, $h(\boldsymbol{x}_i)$.

Outliers

• Many definitions: Atypical observations, extreme values, conditional unusual values, observations outside the expected relation, etc.

• In general, we call an *outlier* an observation that is numerically different from the data. But, is this observation a "mistake," say a result of measurement error, or part of the (heavy-tailed) distribution?

• In the case of normally distributed data, roughly 1 in 370 data points will deviate from the mean by 3*SD. Suppose T=1,000 and we see **9** data points deviating from the mean by more than 3*SD indicates outliers... Which of the **9** observations can be classified as an outlier?

Problem with outliers: They can affect estimates. For example, with small data sets, one big outlier can seriously affect OLS estimates.

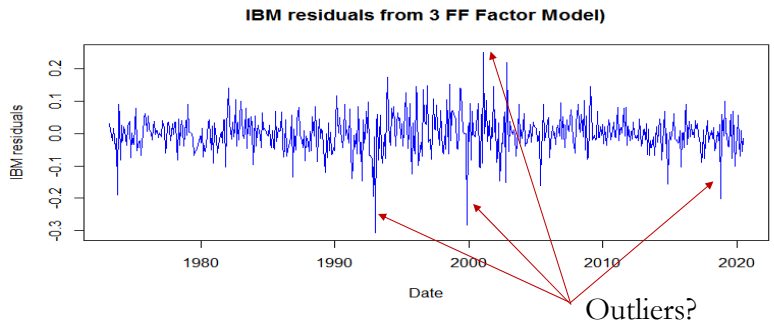
Outliers: Identification

• Informal identification method:

- *Eyeball*: Look at the observations away from a scatter plot.

Example: Plot residuals for the 3 FF factor model for IBM returns

```
x_resid <- residuals(fit_ibm_ff3)
plot(x_resid, typ ="l", col="blue", main ="IBM Residuals from 3 FF Factor Model",
xlab="Date", ylab="IBM residuals")
```



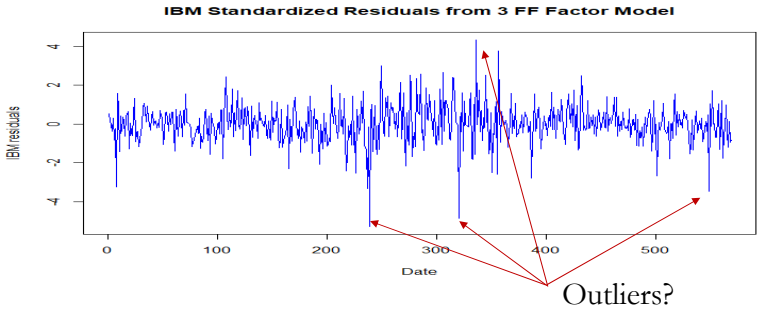
Outliers: Identification

• Formal identifications methods:

- *Standardized residuals*, $e_i/\text{SD}(e_i)$: Check for errors that are 2*SD (or more) away from the expected value.

Example: Plot standardized residuals for IBM residuals

```
x_stand_resid <- x_resid/sd(x_resid)      # standardized residuals
plot(x_stand_resid, typ ="l", col="blue", main ="IBM Standardized Residuals from 3 FF Factor Model", xlab="Date", ylab="IBM residuals")
```



Outliers?

Outliers: Identification – Leverage & Influence

• Formal identifications methods:

- *Leverage statistics*: It measures the difference of an independent data point from its mean. High leverage observations can be potential outliers. Leverage is measured by the diagonal values of the **P** matrix:

$$h_j = 1/T + (x_j - \bar{x})/[(T-1)\, s_x^2].$$

Note: An observation can have high leverage, but no *influence*.

- *Influence statistics:* $Dif\ beta$. It measures how much an observation influences a parameter estimate, say b_j. $Dif\ beta$ is calculated by removing an observation, say i, recalculating b_j, say $b_j(-i)$, taking the difference in betas and standardizing it. Then,

$$Dif\ beta_{j(-i)} = \frac{\sum_{j=1}^{k}(b_j - b_j(-i))}{SE[b_j]}$$

Outliers: Identification – Leverage & Influence

• A related popular influence statistic is *Distance D (as in Cook's D)*. It measures the effect of deleting an observation, say i, on the fitted values, say $\hat{y}_j$. Using the previous notation we have:

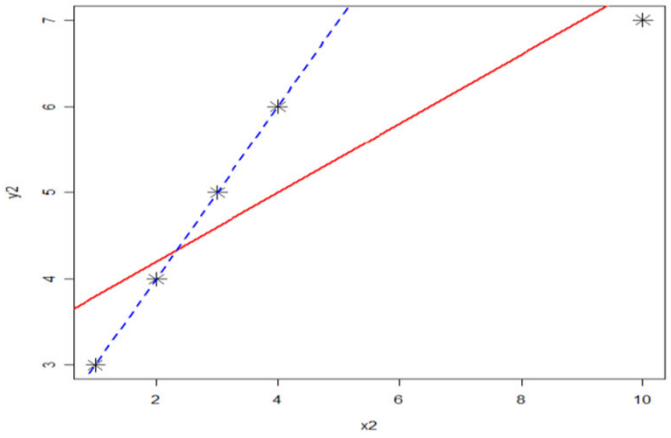
$$D_i = \frac{\sum_{i=1}^{T}(\hat{y}_j - \hat{y}_j(-i))}{k * MSE}$$

where k is the number of parameters in the model and MSE is mean square error of the regression model (MSE = RSS/T).

• The identification statistics are usually compared to some *ad-hoc* cut-off values. For example, for Cook's D, if $D_i > 4/T \Rightarrow$ observation i is considered a (potential) highly influential point.

• The analysis can also be carried out for groups of observations. In this case, we look for blocks of highly influential observations.

Outliers: Leverage & Influence



• Deleting the observation in the upper right corner has a clear effect on the regression line. This observation has *leverage* and *influence*.

Outliers: Summary of Rules of Thumb

• General rules of thumb (ad-hoc thresholds) used to identify outliers:

Measure	**Value**
abs(stand resid)	> 2
leverage	$> (2\,k + 2)/T$
abs($Dif\ beta$)	$> 2/\text{sqrt}(T)$
Cook's D	$> 4/T$

In general, if we have 5% or less observations exceeding the ad-hoc thresholds, we tend to think that the data is OK.

Outliers: Example

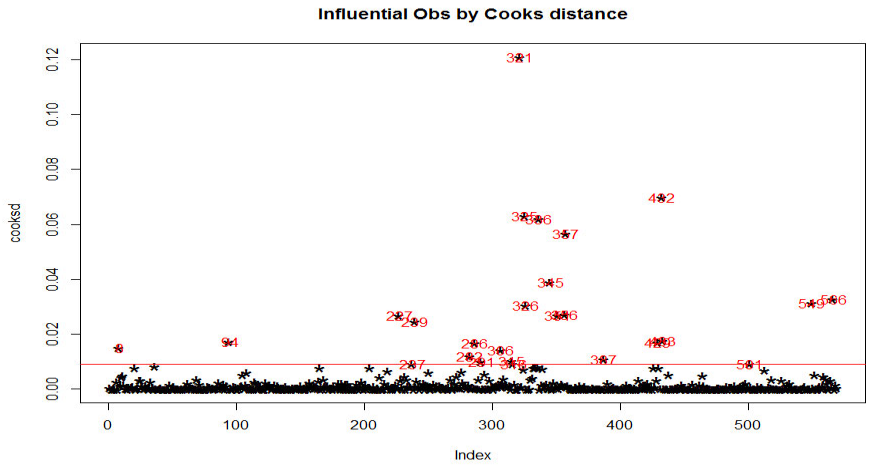
Example: Cook's D for IBM returns using the 3 FF Factor Model

```
y <- ibm_x
x <- cbind(x0, Mkt_RF, SMB, HML)
dat_xy <- data.frame(y, x)
fit_ibm_ff3 <- lm(y ~ x - 1)
cooksd <- cooks.distance(fit_ibm_ff3)
# plot cook's distance
plot(cooksd, pch="*", cex=2, main="Influential Obs by Cooks distance")
# add cutoff line
abline(h = 4*mean(cooksd, na.rm=T), col="red")  # add cutoff line
# add labels
text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>4*mean(cooksd,
na.rm=T), names(cooksd),""), col="red")  # add labels

 # influential row numbers
influential <- as.numeric(names(cooksd)[(cooksd > 4*mean(cooksd, na.rm=T))])
 # print first 10 influential observations.
head(dat_xy[influential, ], n=10L)
```

Outliers: Example

Example (continuation): Cook's D for IBM (3 Factor-Model)



Outliers: Example

Example (continuation): Cook's D for IBM (3 Factor-Model)

> # print first 10 influential observations.
>head(**dat_xy**[influential,], n=10L)

	y	V1	Mkt_RF	SMB	HML
8	-0.16095068	1	0.0475	0.0294	0.0219
94	0.01266444	1	0.0959	-0.0345	-0.0835
227	-0.04237227	1	0.1084	-0.0224	-0.0403
237	-0.19083575	1	0.0102	0.0205	-0.0210
239	-0.30648638	1	0.0153	0.0164	0.0252
282	0.07787100	1	-0.0597	-0.0383	0.0445
286	0.20734626	1	0.0625	-0.0389	0.0117
291	0.15218986	1	0.0404	-0.0565	-0.0006
306	0.13928315	1	-0.0246	-0.0512	-0.0096
315	0.16196934	1	0.0433	0.0400	0.0253

Note: There are easier ways to plot Cook's D and identify the suspect outliers. The package *olsrr* can be used for this purpose too.

Outliers: Example

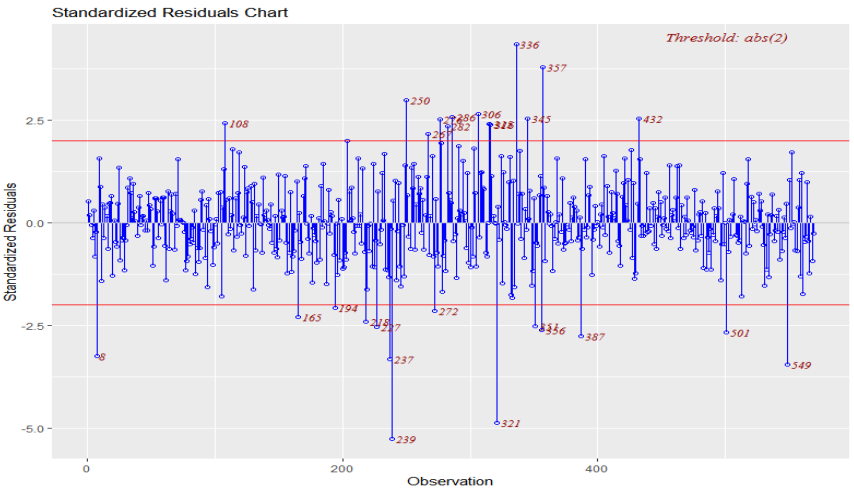
Example: Different tools to check for outliers for IBM returns
We will use the package *olsrr* --install it with **install.packages()**.
install.packages("olsrr")

```
library(olsrr)                                  # need to install package olsrr
x_resid <- residuals(fit_ibm_ff3)
x_stand_resid <- x_resid/sd(x_resid)            # standardized residuals
sum(x_stand_resid > 2)                          # Rule of thumb count (5% count is OK)
x_lev <- ols_leverage(fit_ibm_ff3)              # leverage residuals
sum(x_lev > (2*k+2)/T)                          # Rule of thumb count (5% count is OK)
sum(cooksd > 4/T)                               # Rule of thumb count (5% count is OK)
ols_plot_resid_stand(fit_ibm_ff3)               # Plot standardized residuals
ols_plot_cooksd_bar(fit_ibm_ff3)                # Plot Cook's D measure
ols_plot_dffits(fit_ibm_ff3)                    # Plot Difference in fits
ols_plot_dfbetas(fit_ibm_ff3)                   # Plot Difference in betas

> sum(x_stand_resid > 2)
[1] 13                                          # 5%? = 13/569 = 0.0228
> sum(x_lev > (2*k+2)/T)
[1] 32                                          # 5%? = 32/569 = 0.0562
> sum(cooksd > 4/T)
[1] 38                                          # 5%? = 38/569 = 0.0668
```

Outliers: Example

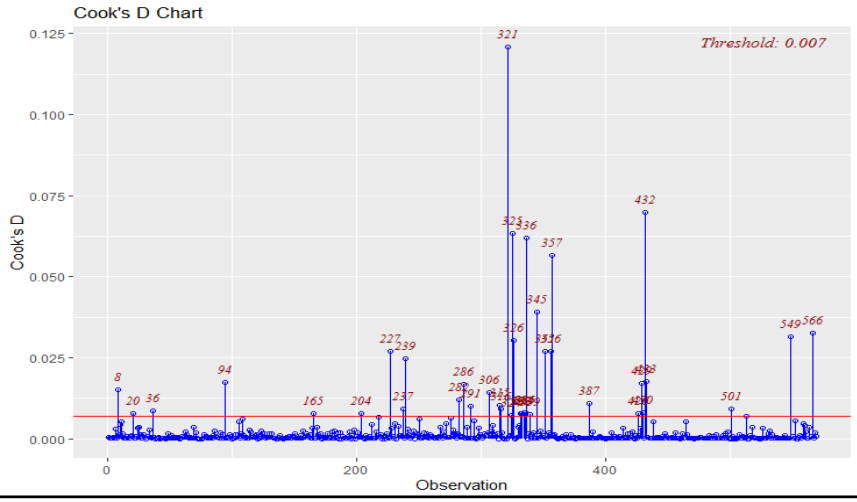
Example (continuation):
>ols_plot_resid_stand(**fit_ibm_ff3**) # Plot Standardize residuals



Outliers: Example

Example (continuation):

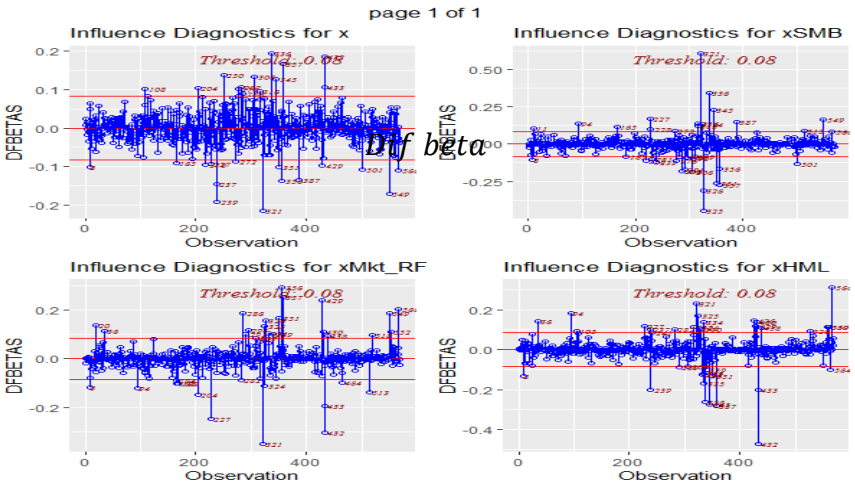
>ols_plot_cooksd_bar(fit_ibm_ff3) # Plot Cook's D measure



Outliers: Example

Example (continuation):

>ols_plot_dfbetas(fit_ibm_ff3)



Outliers: Application – Rules of Thumb

• The histogram, Boxplot, and quantiles helps us see some potential outliers, but we cannot see which observations are potential outliers. For these, we can use Cook's D, $Dif\ beta$'s, standardized residuals and leverage statistics, which are estimated for each i.

Observation Type	Proportion	Cutoff	
Outlier	**0.0228**	**2.0000**	(abs(standardized residuals) > 2)
Outlier	0.1474	$2/\text{sqrt}(T)$	(diffit > 2/sqrt(1038) = **0.0621**)
Outlier	**0.0668**	$4/T$	(cookd > 4/1038 = **0.00385**)
Leverage	**0.0562**	$(2\ k+2)/T$	(h=leverage > **.00771**)

Outliers: What to do?

• Typical solutions:

- Use a non-linear formulation or apply a transformation (log, square root, etc.) to the data.

- Remove suspected observations. (Sometimes, there are theoretical reasons to remove suspect observations. Typical procedure in finance: remove public utilities or financial firms from the analysis.)

- Winsorization of the data (cut an α% of the highest and lowest observations of the sample).

- Use dummy variables.

- Use LAD (quantile) regressions, which are less sensitive to outliers.

- Weight observations by size of residuals or variance (robust estimation).

• General rule: Present results with or without outliers.

Multicollinearity

• The **X** matrix is *singular* (perfect collinearity) or *near singular (multicollinearity)*.

• *Perfect collinearity*

Not much we can do. OLS will not work ⇒ **X'X** cannot be inverted. The model needs to be reformulated.

• *Multicollinearity*.

OLS will work. **β** is still unbiased. The problem is in $(\mathbf{X'X})^{-1}$; that is, in the Var[**b** | **X**]. Let's see the effect on the variance of particular coefficient, b_k.

Recall the estimated Var[b_k | **X**] is the *k*th diagonal element of $\sigma^2(\mathbf{X'X})^{-1}$.

Multicollinearity & VIF

• Let define $R^2_{k.}$ as the R^2 in the regression of $\boldsymbol{x}_k$ on the other regressors, $\mathbf{X}_k$. Then, we can show the estimated Var[b_k | **X**] is

$$\text{Var}[b_k \mid \mathbf{X}] = \frac{s^2}{[(1-R^2_{k.})\sum_{i=1}^{n}(x_{ik}-\bar{x}_k)^2]}.$$

⇒ the higher $R^2_{k.}$ –i.e., the fit between $\boldsymbol{x}_k$ and the rest of the regressors–, the higher Var[b_k | **X**].

• The ratio $\frac{1}{(1-R^2_{k.})}$ is called the Variance Inflation Factor of regressor k, or VIF_k. It should be equal to 1 when $\boldsymbol{x}_k$ is unrelated to the rest of the regressors (including a constant). The higher it is, the higher the linear correlation between $\boldsymbol{x}_k$ and the rest of the regressors.

• A common rule of thumb: If $\text{VIF}_k > 5$, concern.

Multicollinearity: Signs

• Signs of Multicollinearity:

- Small changes in **X** produce wild swings in **b**.
- High R^2, but **b** has low t-values –i.e., high standard errors
- "Wrong signs" or difficult to believe magnitudes in **b**.

• There is no *cure* for collinearity. Estimating something else is not helpful; for example, transforming variables to eliminate multicollinearity, since we are interested in the effect of X on y, not necessarily the effect of f(X) on g(y).

Multicollinearity: VIF and Condition Index

• Popular measures to detect multicollinearity:

- VIF
- Condition number (based on singular values), or *K#*.

• Belsley (1991) proposes to calculate VIF and the condition number, using R_X, the correlation matrix of the standardized regressors:

$$VIF_k = \text{diag}(R_X^{-1})_k$$

$$\text{Condition Index} = \varkappa_k = \text{sqrt}(\lambda_1 / \lambda_k)$$

where $\lambda_1 > \lambda_2 > ... > \lambda_p > ...$ are the ordered eigenvalues of R_X.

• Belsley's (1991) rules of thumb for $\varkappa_k$:

- below 10 ⇒ good
- from 10 to 30 ⇒ concern
- greater than 30 ⇒ trouble (>100, a disaster!)

Multicollinearity: Example

Example: Check for multicollinearity for IBM returns 3-factor model

```
library(olsrr)
ols_vif_tol(fit_ibm_ff3)
ols_eigen_cindex(fit_ibm_ff3)
```

```
> ols_vif_tol(fit_ibm_ff3)
  Variables         Tolerance      VIF
1   xMkt_RF         0.8901229 1.123440
2   xSMB            0.9147320 1.093216
3   xHML            0.9349904 1.069530
> ols_eigen_cindex(fit_ibm_ff3)
  Eigenvalue Condition Index  intercept      xMkt_RF       xSMB       xHML
1  1.4506645       1.000000 0.01557614 0.24313961 0.212001760 0.1518949
2  1.0692689       1.164770 0.66799183 0.01432250 0.001789253 0.2129328
3  0.7967889       1.349310 0.16184731 0.01239755 0.576432492 0.4107435
4  0.6832777       1.457085 0.15458473 0.73014033 0.209776495 0.2244287
```

Note: Multicollinearity does not seem to be a problem.

Multicollinearity: Remarks

- Best approach: Recognize the problem and understand its implications for estimation.

Note: Unless we are very lucky, some degree of multicollinearity will always exist in the data. The issue is: when does it become a problem?