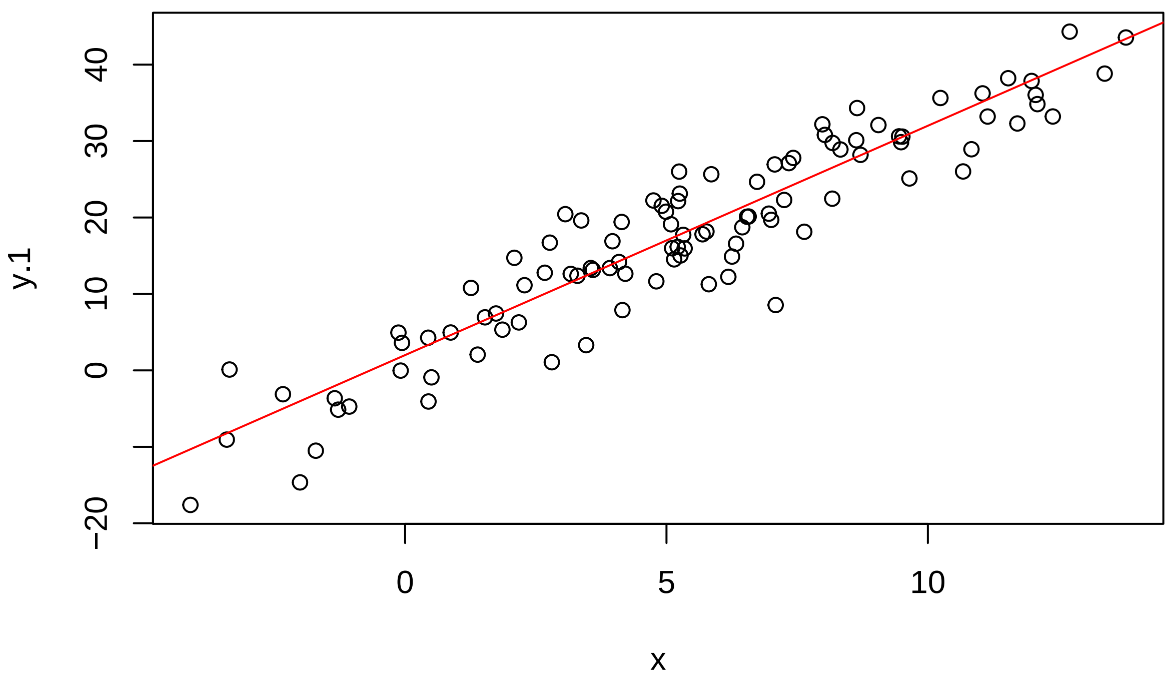
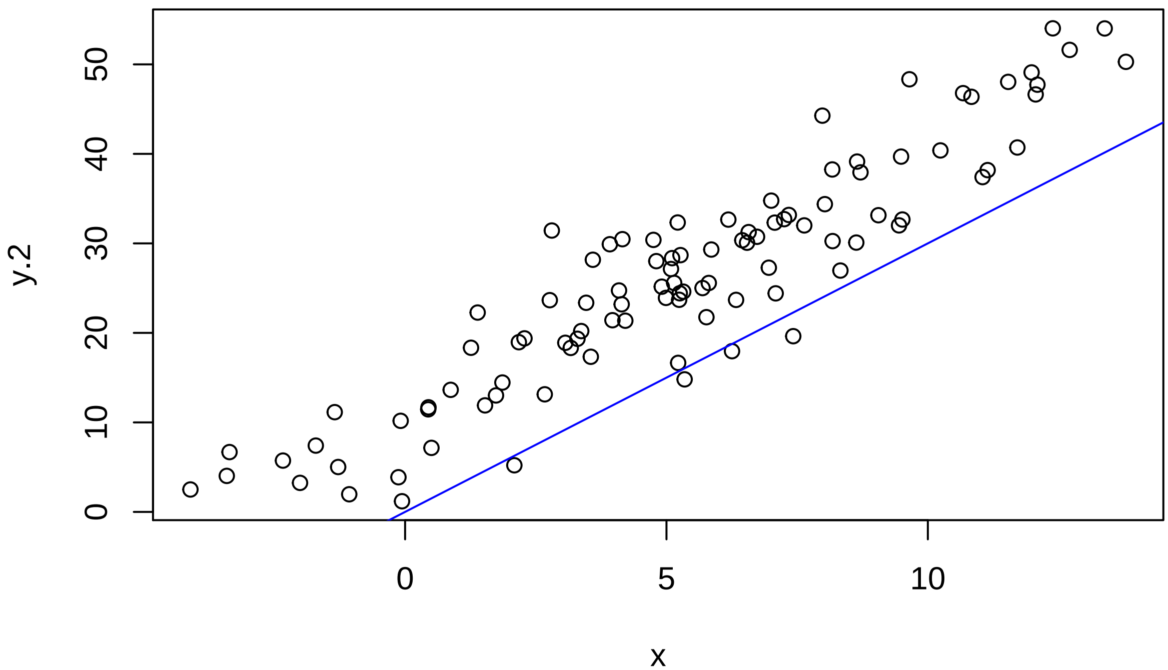
Comp stats notebook

Using simple linear regression to prove the properties of LS estimator

```r
## generating fake dataset
set.seed(21)
x <- rnorm(n=100, mean=5, sd=4)
# the regression lines in the following two models should
# be the same since the error mean is absorbed in the model intercept
eps.1 <- rnorm(n=100, mean=0, sd=1)
eps.2 <- rnorm(n=100, mean=2, sd=1)
y.1 <- 2 + 3*x + 5*eps.1
y.2 <- 3*x + 5*eps.2
plot(x,y.1)
abline(a=2,b=3, col="red")
```




```r
plot(x,y.2)
abline(a=0, b=3, col="blue")
```



Now let's build the regression model.

```
reg.1 <- lm(y.1~x)
reg.2 <- lm(y.2~x)
```

```
summary(reg.1)
```

```
##
## Call:
## lm(formula = y.1 ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.1965  -2.8125   0.3336   3.4590   9.7342
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.5335     0.7624   2.011    0.047 *
## x             2.9921     0.1140  26.238   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.662 on 98 degrees of freedom
## Multiple R-squared:  0.8754, Adjusted R-squared:  0.8741
## F-statistic: 688.5 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
summary(reg.2)
```

```
##
## Call:
## lm(formula = y.2 ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.6418  -2.7347   0.3618   3.4257  12.6856
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.5327     0.8038   13.10   <2e-16 ***
## x             2.9281     0.1202   24.35   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.915 on 98 degrees of freedom
## Multiple R-squared:  0.8582, Adjusted R-squared:  0.8568
## F-statistic: 593.1 on 1 and 98 DF,  p-value: < 2.2e-16
```

What if we want to compute the coefficients by hand? Let's do it!

```r
X <- matrix(cbind(rep(1,100), x), nrow = 100, ncol = 2)
xtx.inv <- solve(t(X)%*%X)
# coefficients
print("Beta 1")
```

```
## [1] "Beta 1"
```

```r
(beta.1 <- xtx.inv %*% t(X) %*% y.1)
```

```
##          [,1]
## [1,] 1.533473
## [2,] 2.992113
```

```r
print("Beta 2")
```

```
## [1] "Beta 2"
```

```r
(beta.2 <- xtx.inv %*% t(X) %*% y.2)
```

```
##           [,1]
## [1,] 10.532683
## [2,]  2.928132
```

```r
# predictions
y.hat.1 <- X%*%beta.1
y.hat.2 <- X%*%beta.2
# errors
eps.hat.1 <- y.1 - y.hat.1
eps.hat.2 <- y.2 - y.hat.2
# sd estimate
sd.hat.1 <- sum(eps.hat.1**2)/(100-2)
sd.hat.2 <- sum(eps.hat.2**2)/(100-2)
# se for beta
print("Se 1")
```

```
## [1] "Se 1"
```

```r
(se.1 <- sqrt(xtx.inv[2,2]*sd.hat.1))
```

```
## [1] 0.1140356
```

```r
print("Se 2")
```

```
## [1] "Se 2"
```

```r
(se.2 <- sqrt(xtx.inv[2,2]*sd.hat.2))
```

```
## [1] 0.120232
```

```r
# p-values of t-test
tv.1 <- beta.1[2]/(se.1)
tv.2 <- beta.2[2]/(se.2)
print("P-value 1")
```

```
## [1] "P-value 1"
```

```r
(pv.1 <- pt(tv.1, df = 100-2, lower.tail = FALSE))
```

```
## [1] 2.063205e-46
```

```r
print("P-value 2")
```

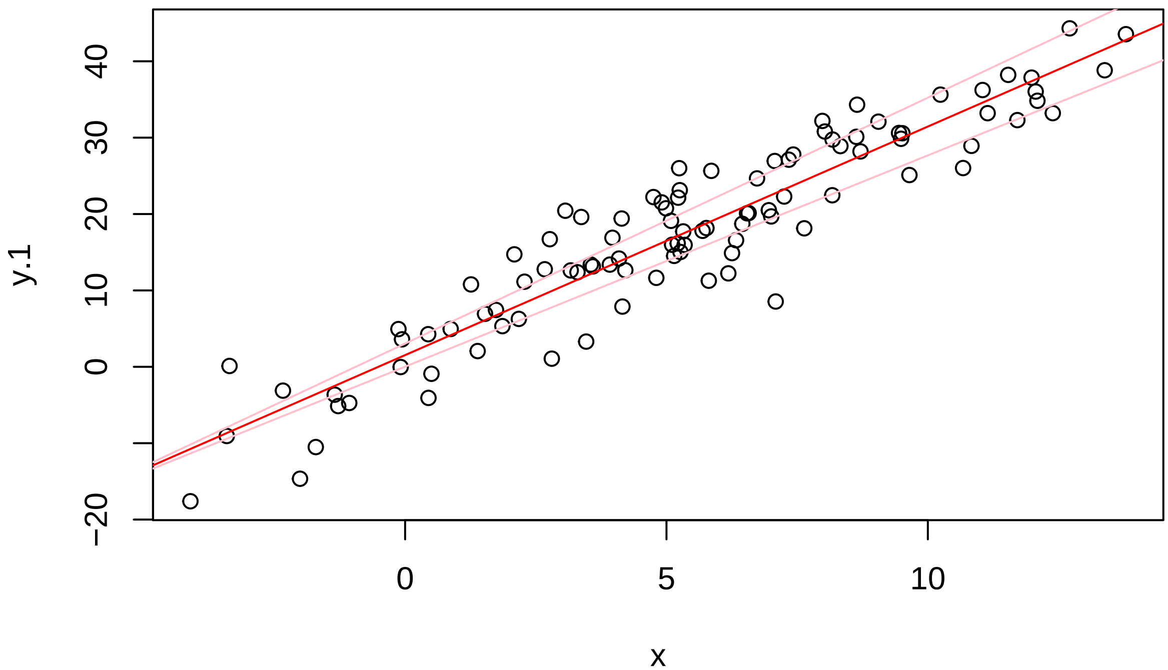
```
## [1] "P-value 2"
```

```r
(pv.2 <- pt(tv.2, df = 100-2, lower.tail = FALSE))
```

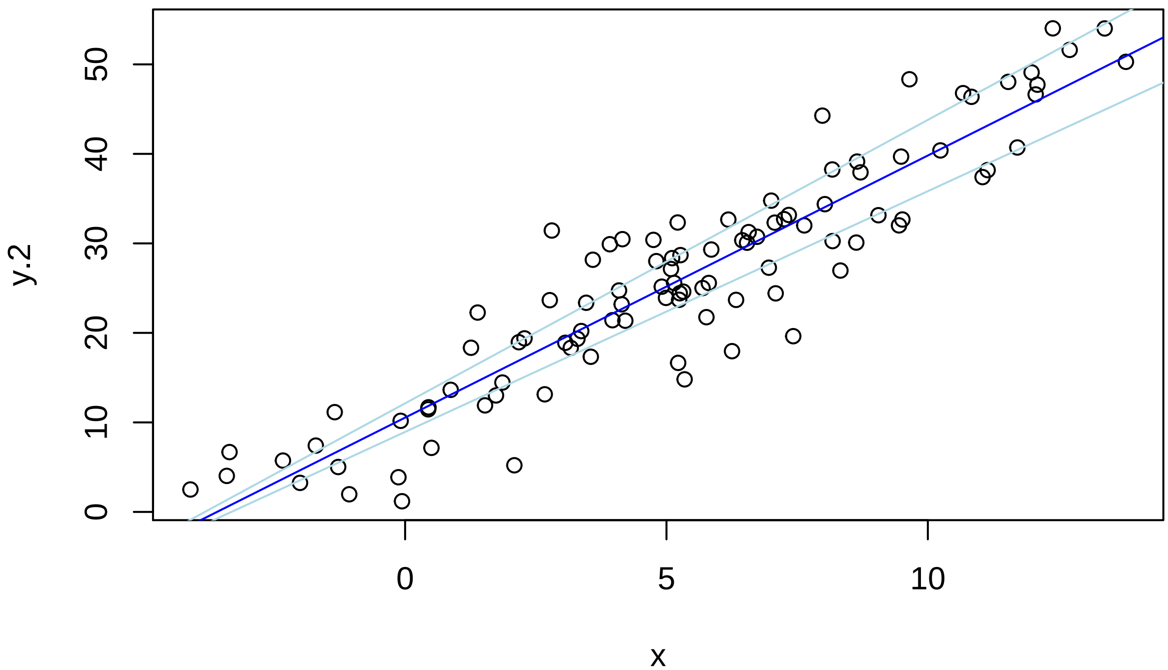
```
## [1] 1.171405e-43
```

Exactly what we have above! What about confidence intervals?

```r
ci.1.95 <- qt(0.975, df=100-2)*se.1
ci.2.95 <- qt(0.975, df=100-2)*se.2

# doing the same procedure for the intercept
se.i.1 <- sqrt(xtx.inv[1,1]*sd.hat.1)
se.i.2 <- sqrt(xtx.inv[1,1]*sd.hat.2)
ci.1.i.95 <- qt(0.975, df=100-2)*se.i.1
ci.2.i.95 <- qt(0.975, df=100-2)*se.i.2



plot(x,y.1)
abline(a=beta.1[1],b=beta.1[2], col="red")
abline(a=beta.1[1]+ci.1.i.95,b=beta.1[2]+ci.1.95, col="pink")
abline(a=beta.1[1]-ci.1.i.95,b=beta.1[2]-ci.1.95, col="pink")
```




```
plot(x,y.2)
abline(a=beta.2[1],b=beta.2[2], col="blue")
abline(a=beta.2[1]+ci.2.i.95,b=beta.2[2]+ci.2.95, col="lightblue")
abline(a=beta.2[1]-ci.2.i.95,b=beta.2[2]-ci.2.95, col="lightblue")
```

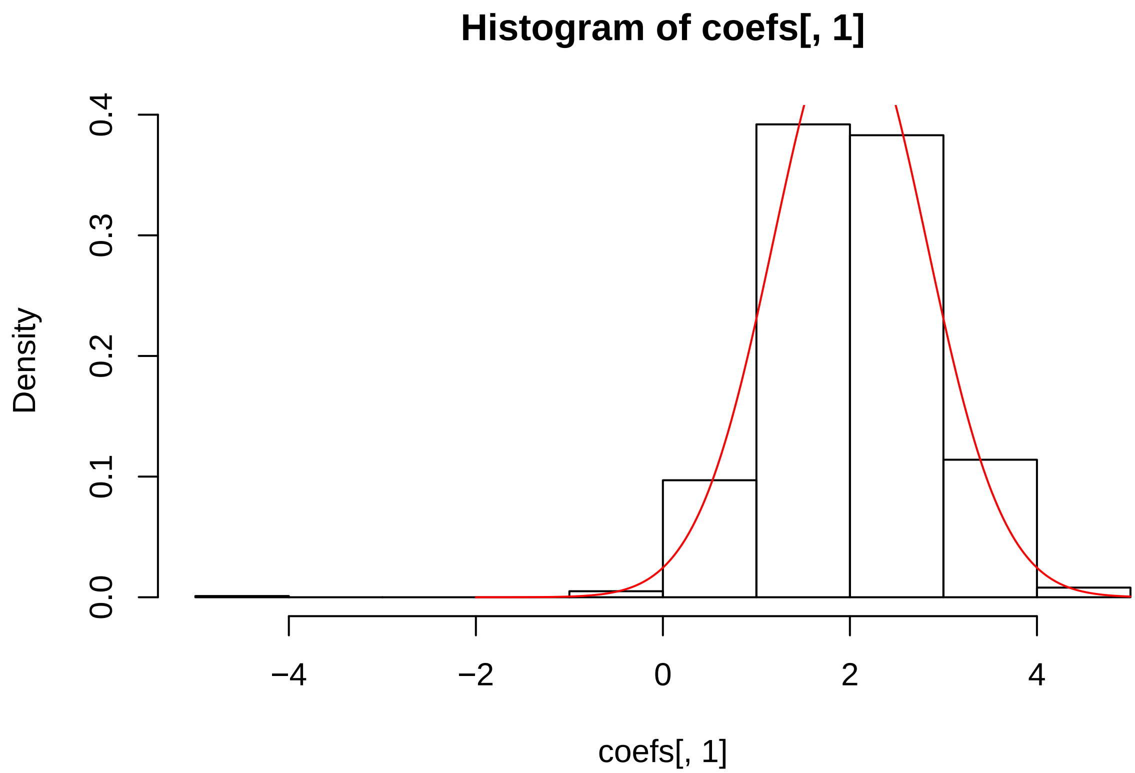


Now let's quickly see how stable this prediction is with a simulation.

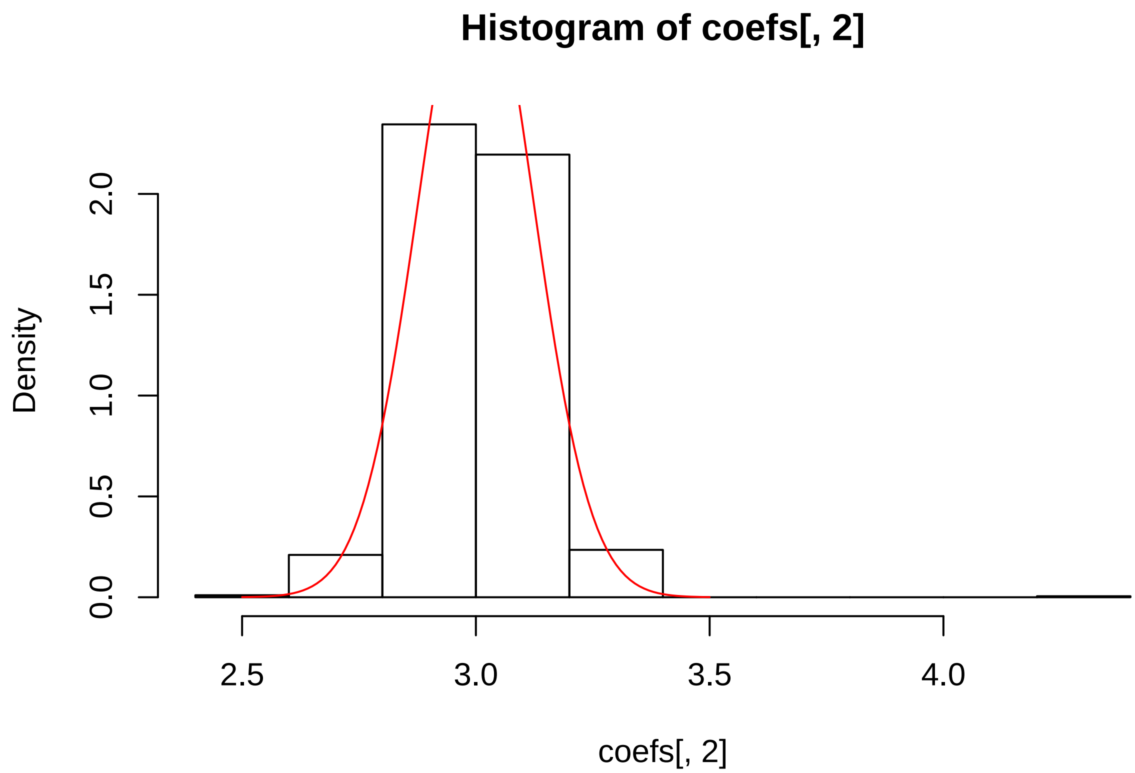
```
## generating fake dataset
set.seed(21)
nsim <- 1000

coefs <- matrix(nrow=nsim, ncol=2)

for(i in 1:nsim){
  eps <- rnorm(n=100, mean=0, sd=1)
  y <- 2 + 3*x + 5*eps
  reg <- lm(y~x)
  beta <- coef(reg)
  names(beta) <- NULL
  coefs[i,] <- beta
}
```

As expected, each coefficient is approximately normally distributed.

```
hist(coefs[,1], freq=FALSE)
lines(seq(-2, 5, by = 0.01), dnorm(seq(-2, 5, by = 0.01), mean= 2, sd = 5*sqrt(xtx.inv[1,1])), col="red
```



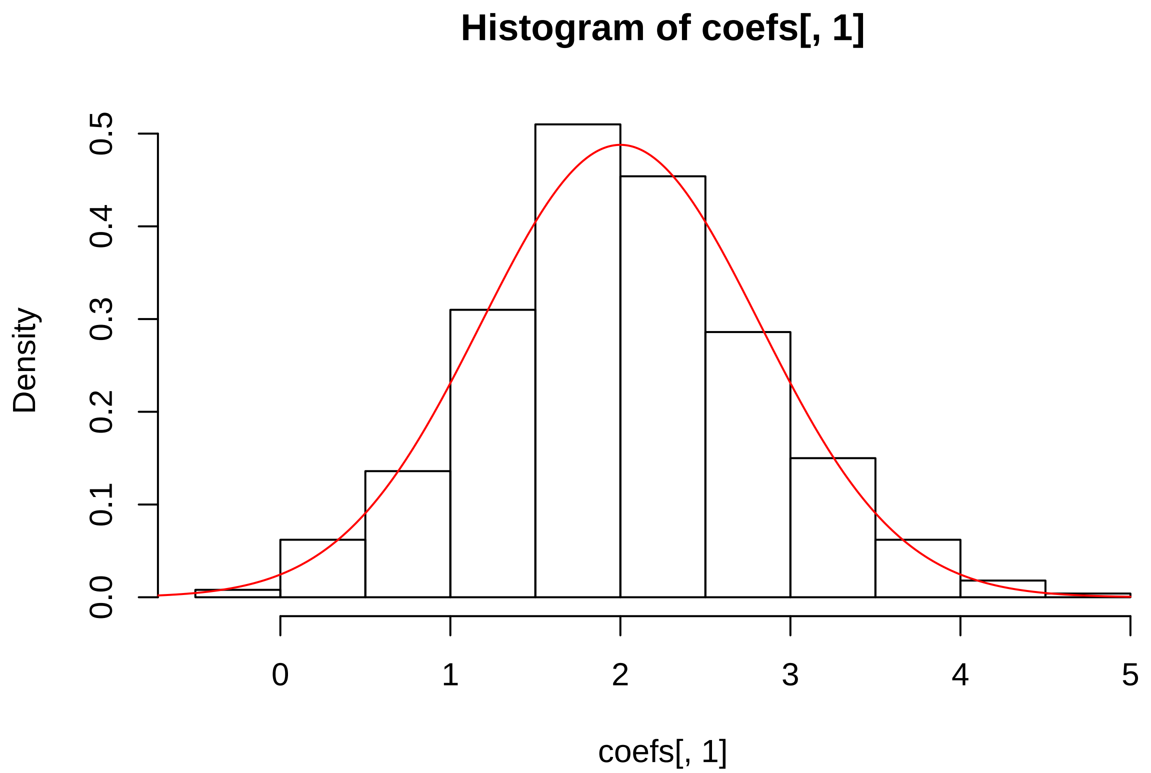

```
hist(coefs[,2], freq = FALSE)
lines(seq(2.5, 3.5, by = 0.01), dnorm(seq(2.5, 3.5, by = 0.01), mean= 3, sd = 5*sqrt(xtx.inv[2,2])),col=
```



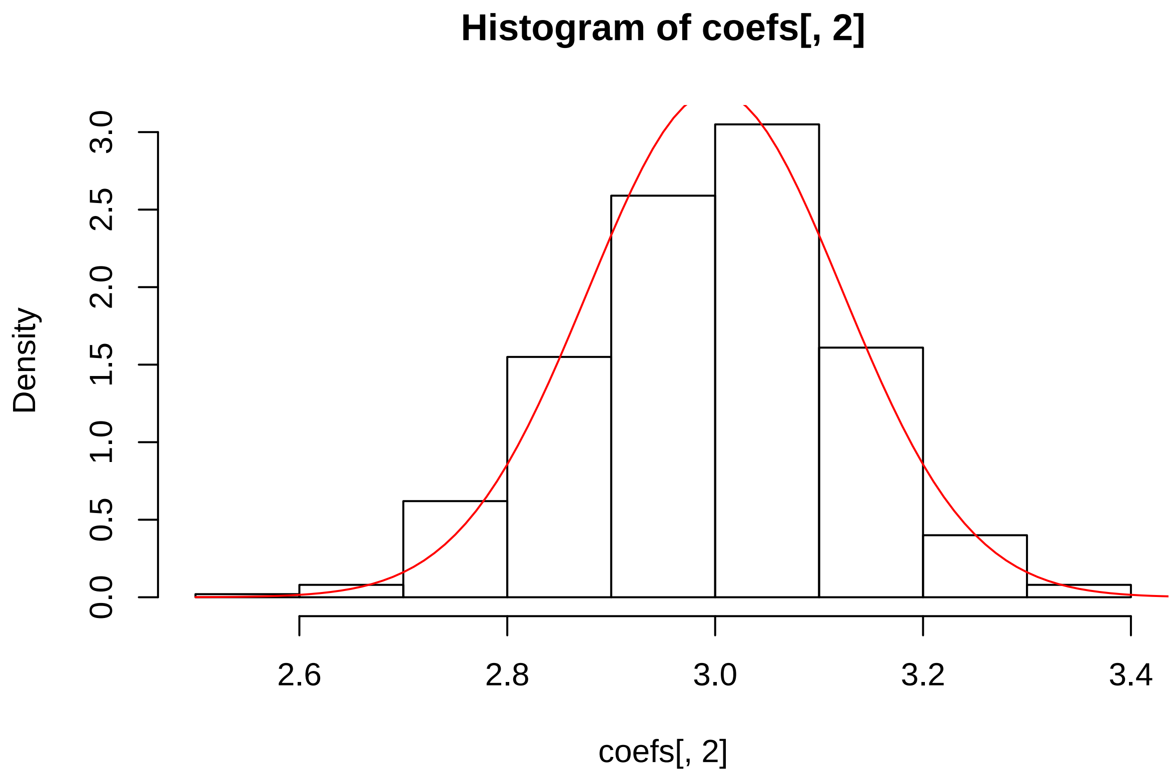
What if X can vary as well?

```
set.seed(21)
nsim <- 1000

coefs <- matrix(nrow=nsim, ncol=2)

for(i in 1:nsim){
  x<-rnorm(n=100, mean=5, sd=4)
  eps <- rnorm(n=100, mean=0, sd=1)
  y <- 2 + 3*x + 5*eps
  reg <- lm(y~x)
  beta <- coef(reg)
  names(beta) <- NULL
  coefs[i,] <- beta
}
```

```
hist(coefs[,1], freq=FALSE)
lines(seq(-2, 5, by = 0.01), dnorm(seq(-2, 5, by = 0.01), mean= 2, sd = 5*sqrt(xtx.inv[1,1])), col="red
```

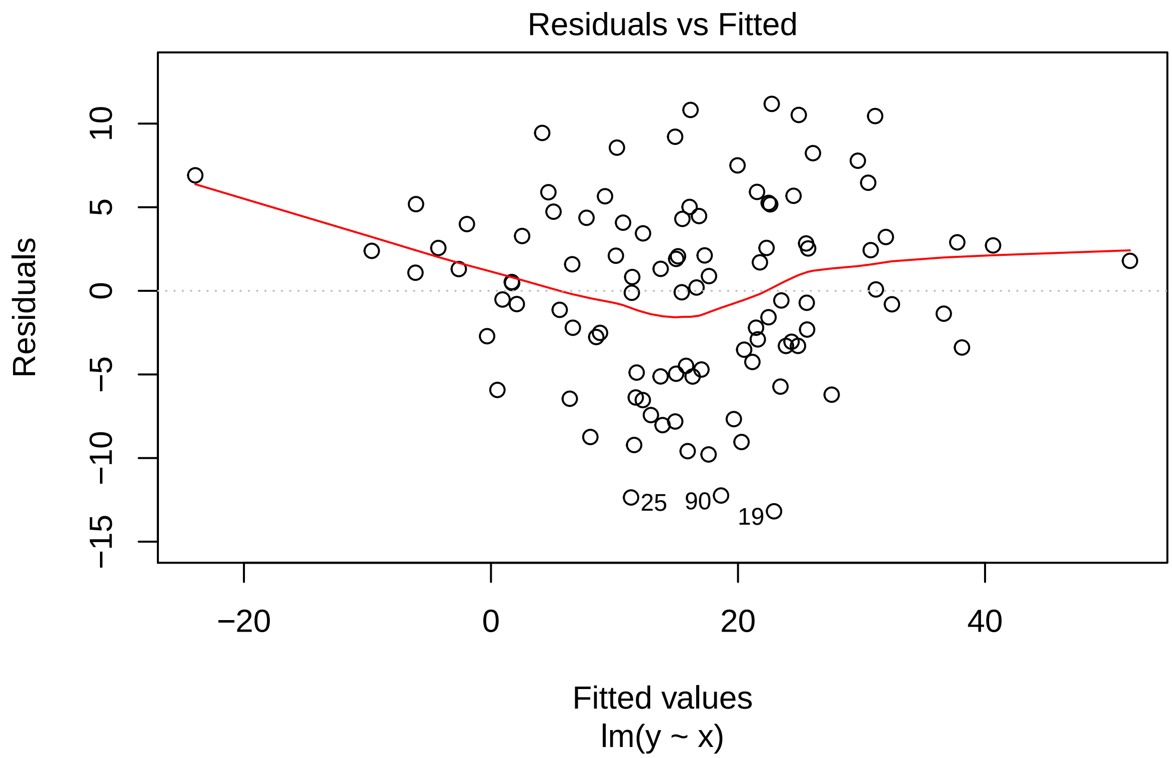


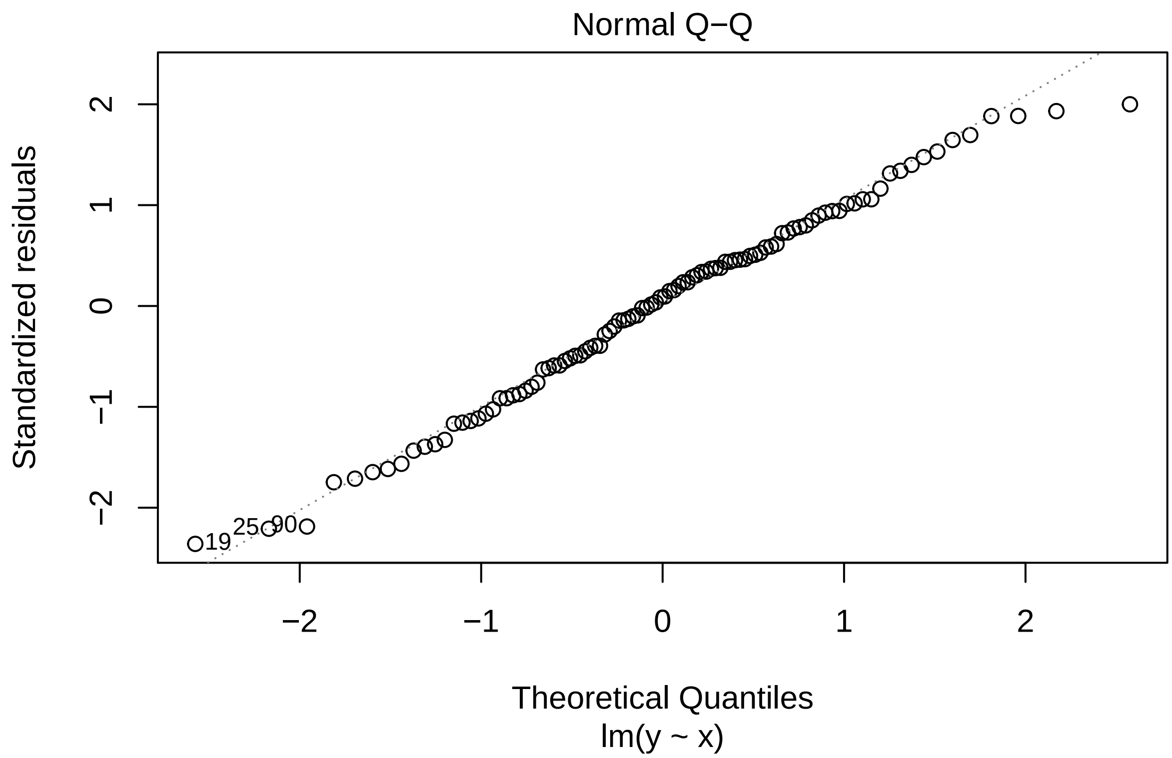

```
hist(coefs[,2], freq = FALSE)
lines(seq(2.5, 3.5, by = 0.01), dnorm(seq(2.5, 3.5, by = 0.01), mean= 3, sd = 5*sqrt(xtx.inv[2,2])),col=
```

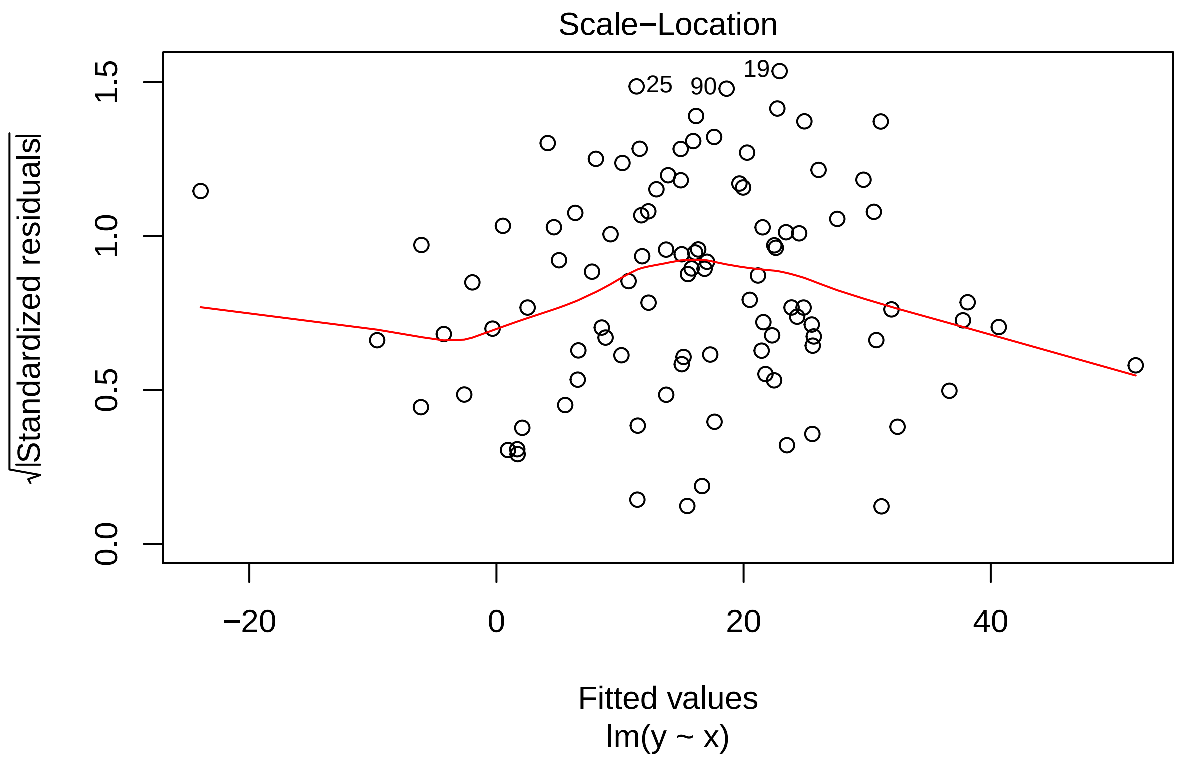


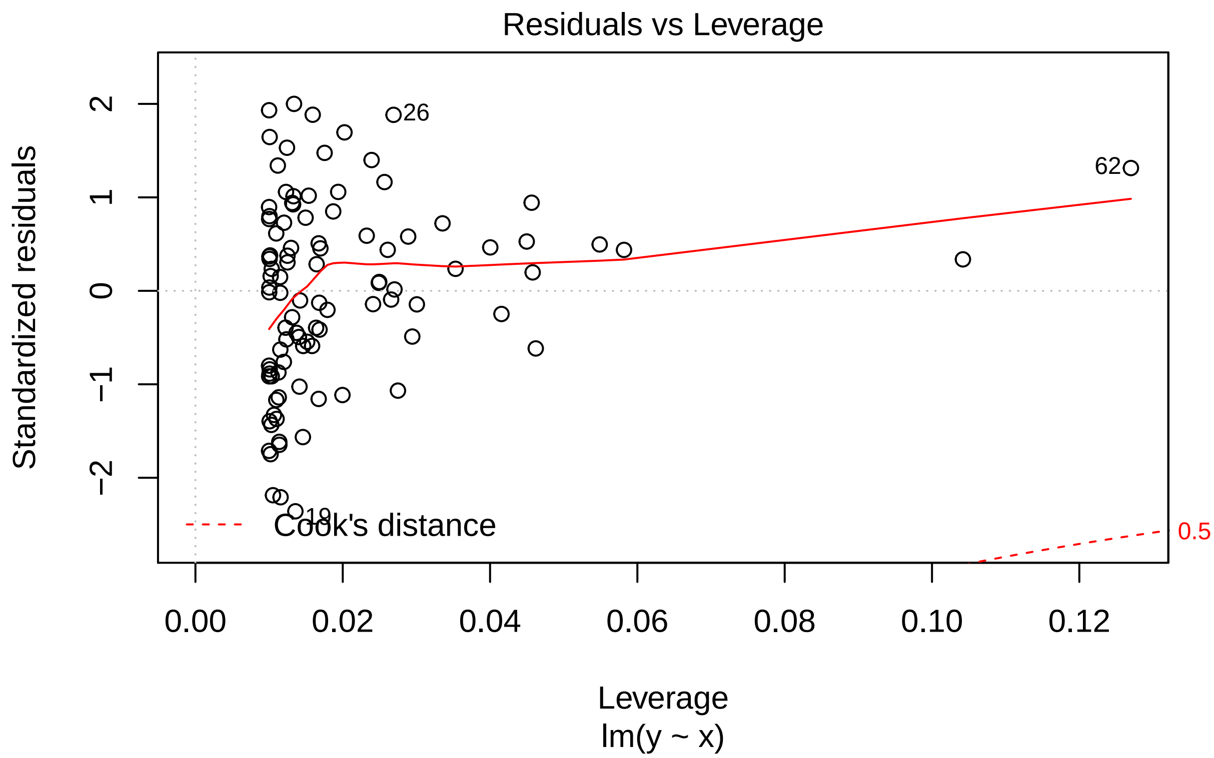
Let's take the last model and test the assumptions!

```
plot(reg)
```


Residuals vs Fitted
Residuals
10
5
0
−5
−10
−15
25
90
19
−20
0
20
40
Fitted values
lm(y ~ x)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
19
25
90
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
25
90
19
−20
0
20
40
Fitted values
lm(y ~ x)

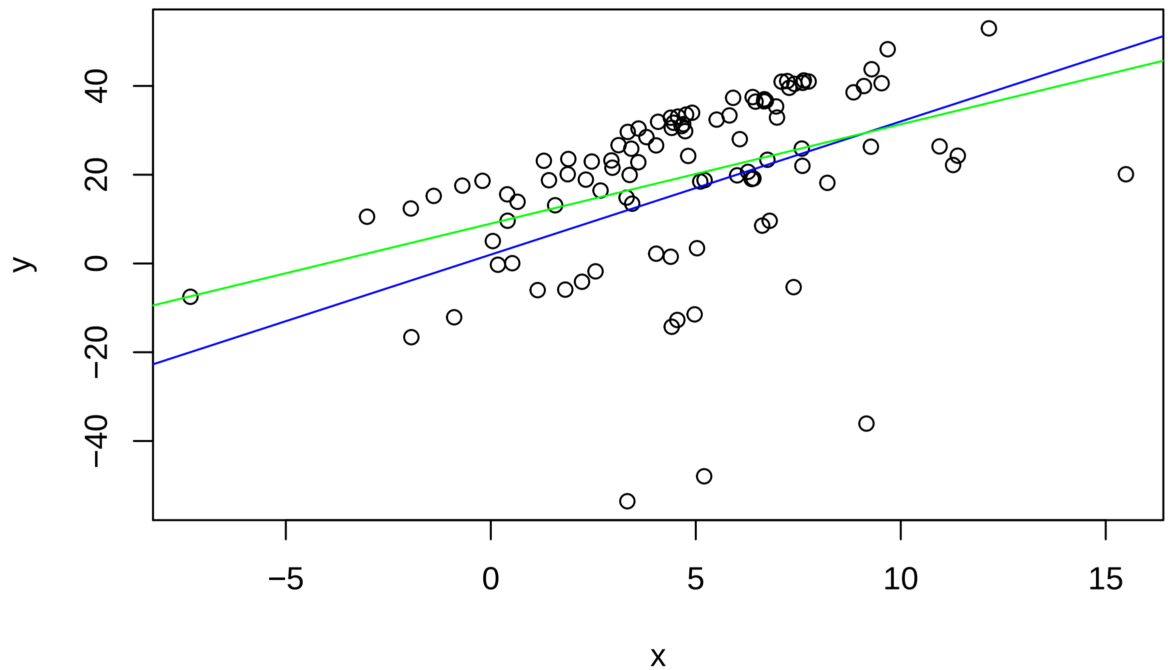


Now let's break some assumptions.

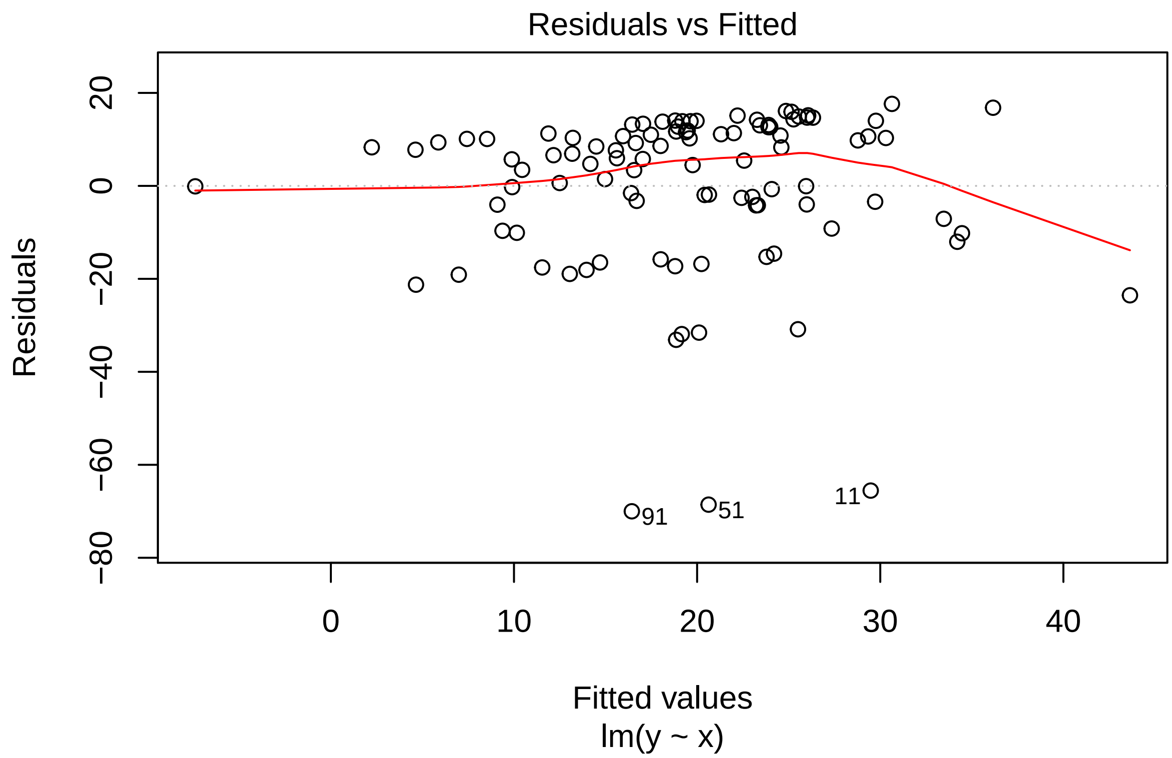
```
# non normality of the noise
eps <- 5 * (1 - rchisq(40, df = 1)) / sqrt(2)
y <- 2 + 3*x + 5*eps
```

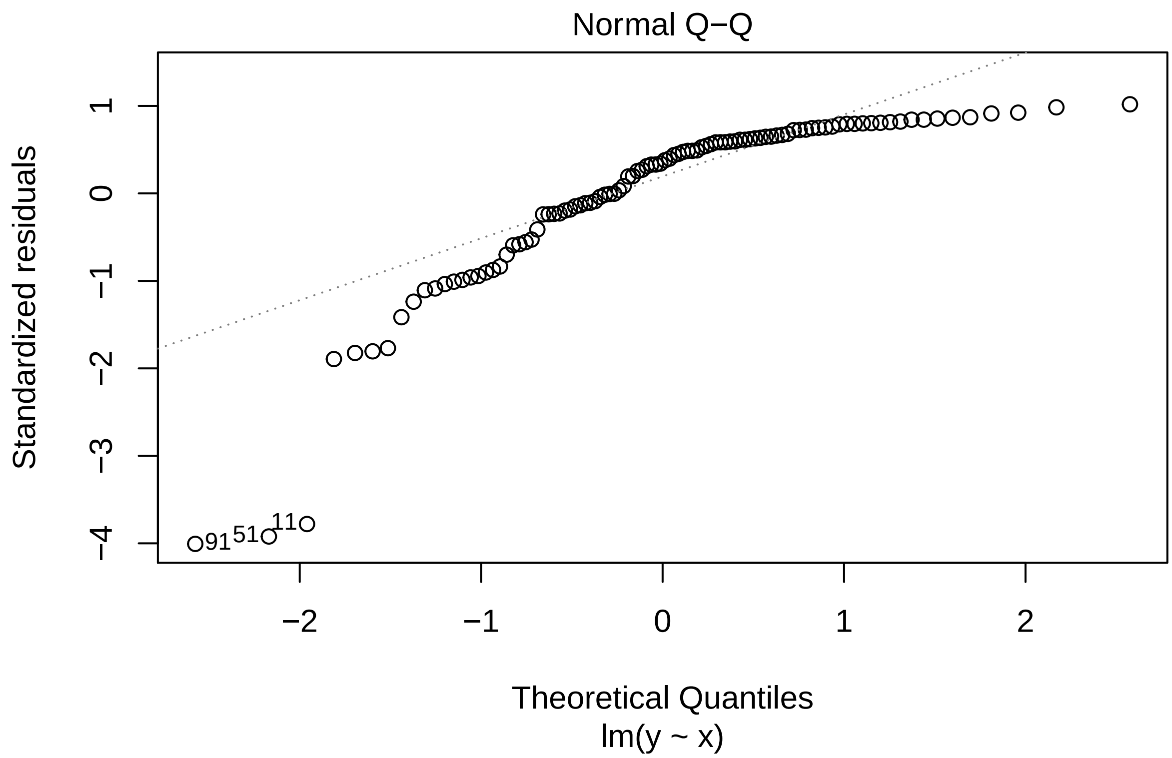
```
## Warning in 2 + 3 * x + 5 * eps: longer object length is not a multiple of
## shorter object length
```

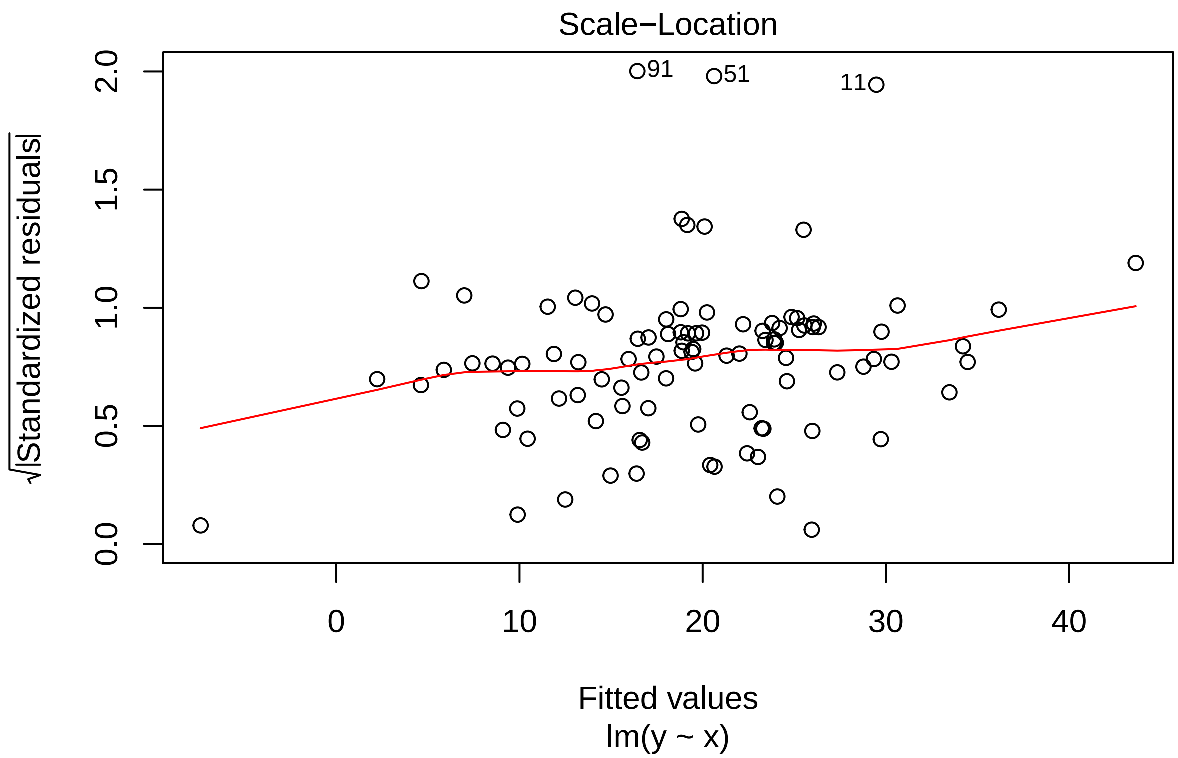
```
plot(x,y)
abline(a=2, b=3, col="blue")
reg <- lm(y~x)
beta <- coef(reg)
names(beta) <- NULL
abline(a=beta[1], b=beta[2], col="green")
```



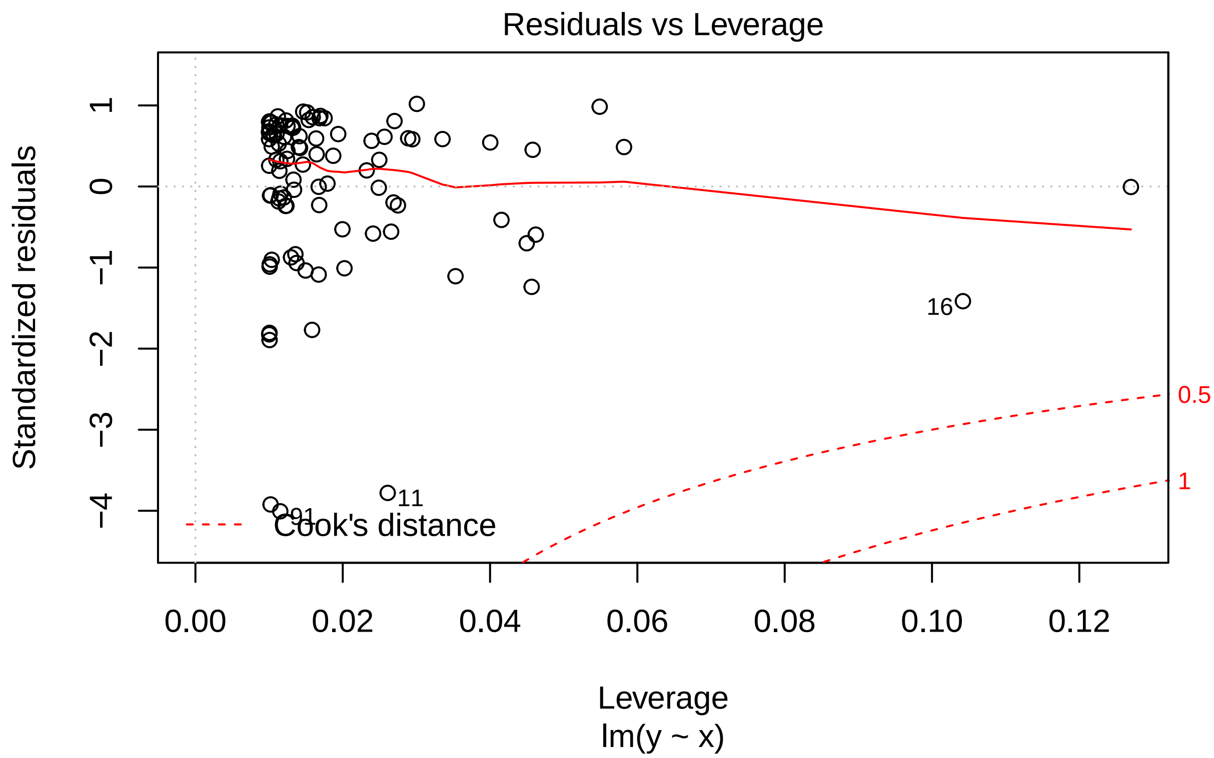

```r
plot(reg)
```


Residuals vs Fitted
Residuals
20
0
−20
−40
−60
−80
0
10
20
30
40
Fitted values
lm(y ~ x)
91
51
11


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
91
51
11
1
0
−1
−2
−3
−4
−2
−1
0
1
2



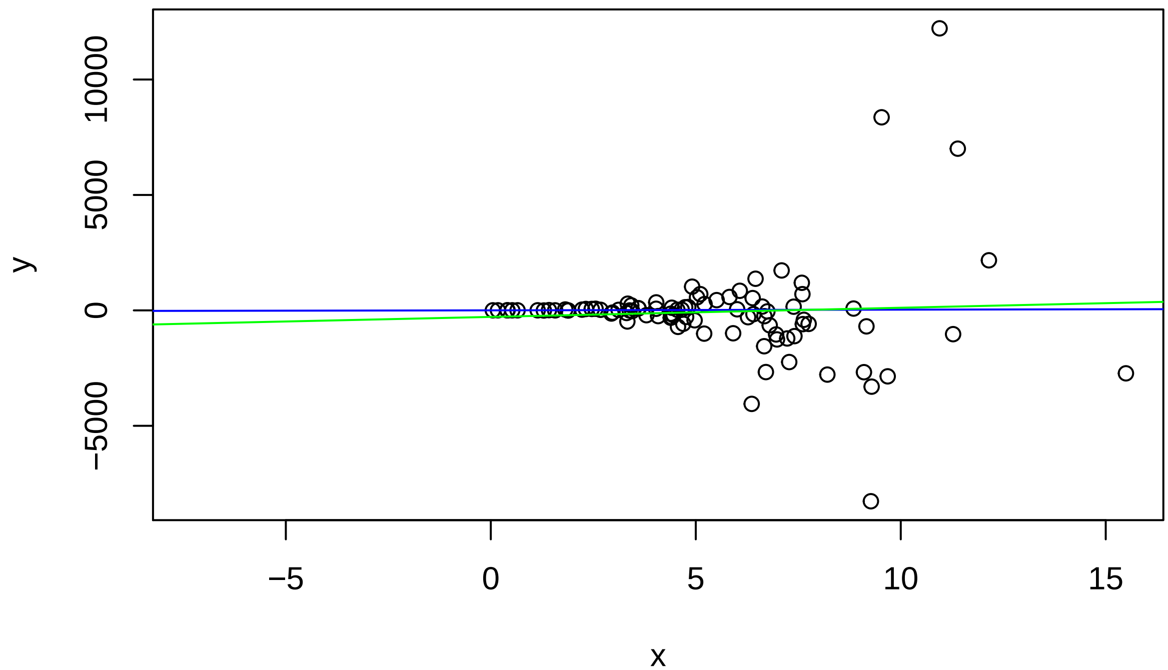
Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
91
51
11
0
10
20
30
40
Fitted values
lm(y ~ x)



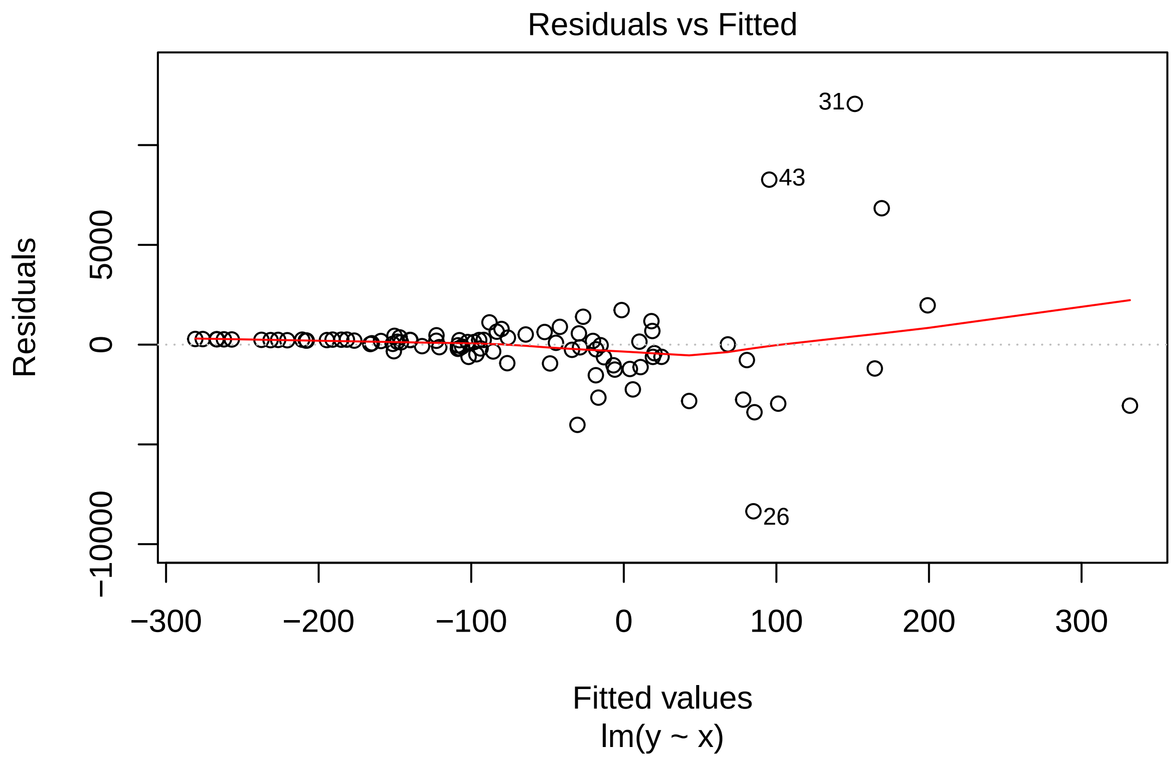

```
# heteroskedasticity
eps <- rnorm(n=100, mean=0, sd= x**3)
```

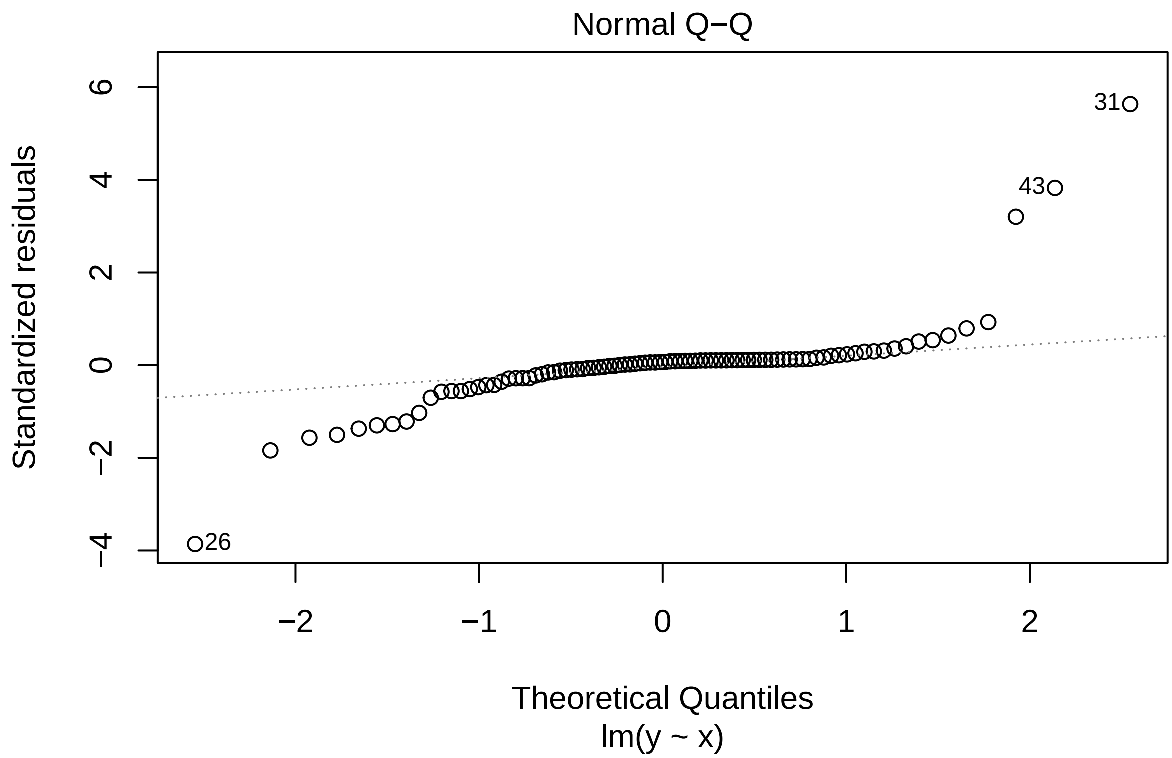
```
## Warning in rnorm(n = 100, mean = 0, sd = x^3): NAs produced
```

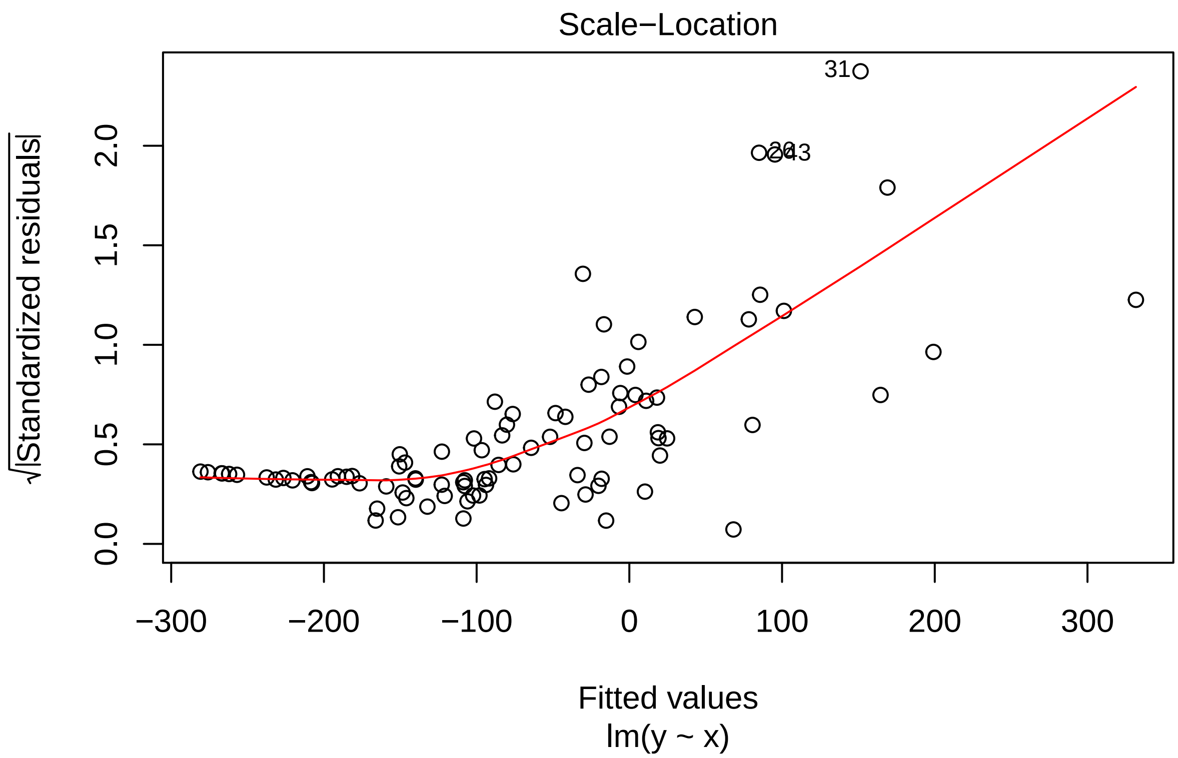
```
y <- 2 + 3*x + 5*eps
plot(x,y)
abline(a=2, b=3, col="blue")
reg <- lm(y~x)
beta <- coef(reg)
names(beta) <- NULL
abline(a=beta[1], b=beta[2], col="green")
```



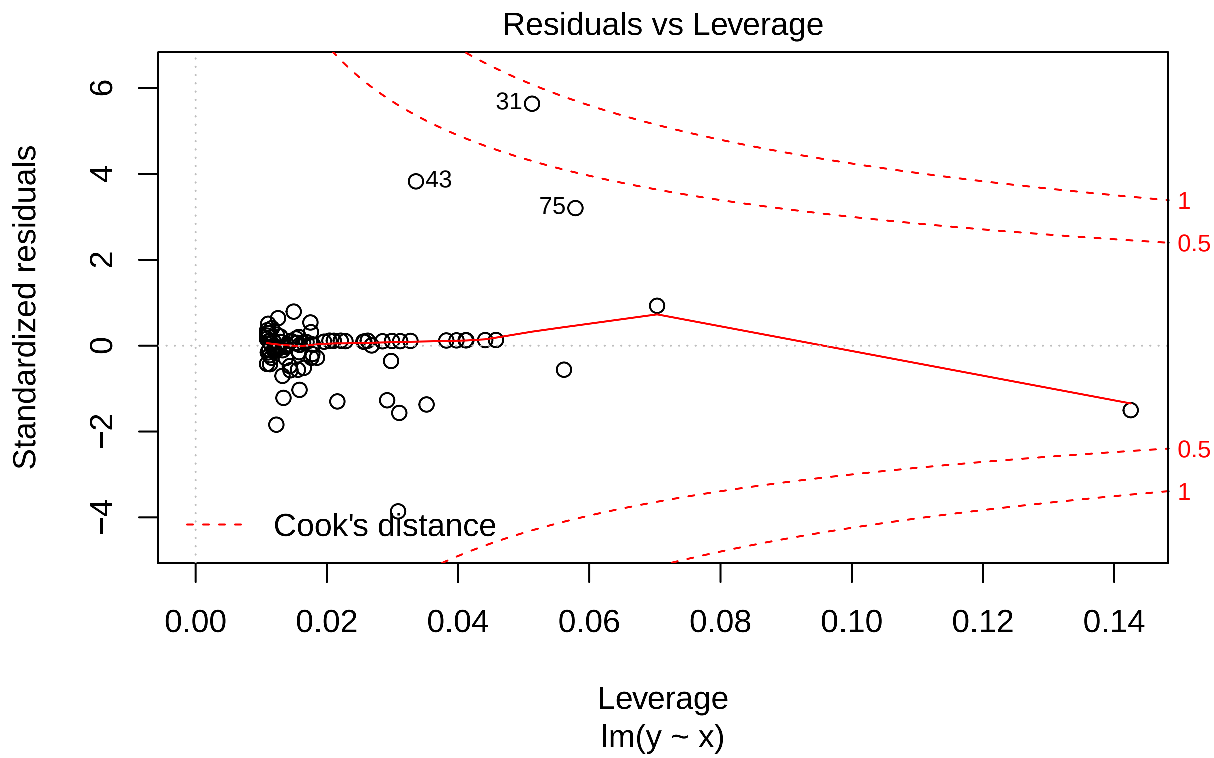

```r
plot(reg)
```


Residuals vs Fitted
Residuals
Fitted values
lm(y ~ x)
31
43
26
5000
0
−10000
−300
−200
−100
0
100
200
300

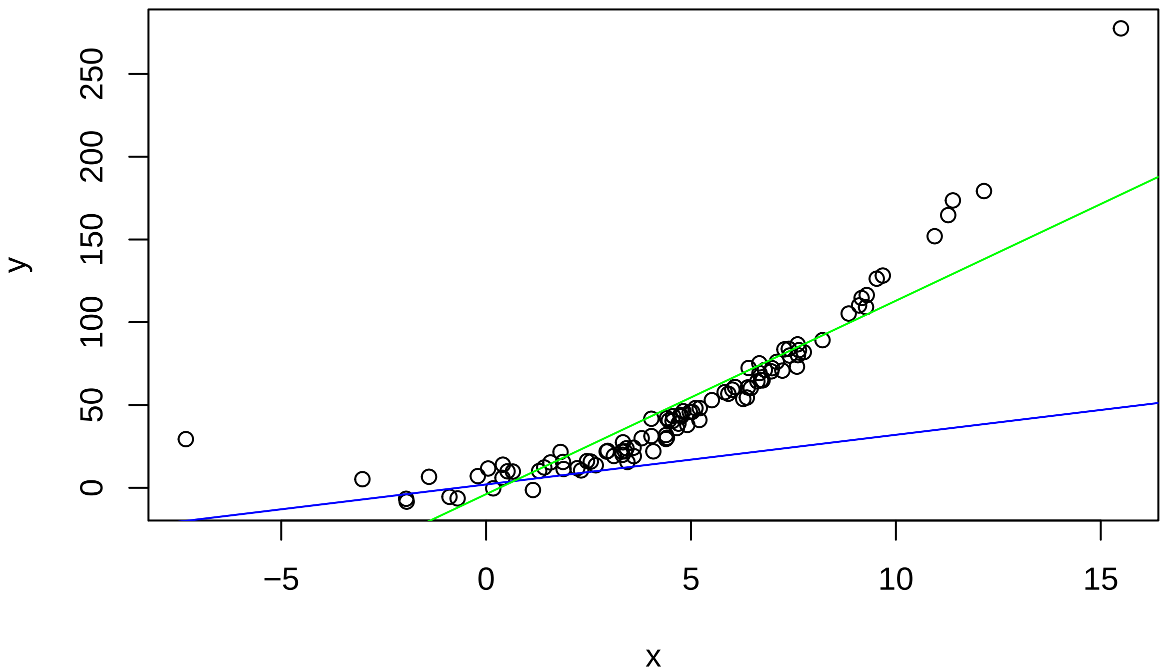

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
−4
−2
0
2
4
6
−2
−1
0
1
2
31
43
26



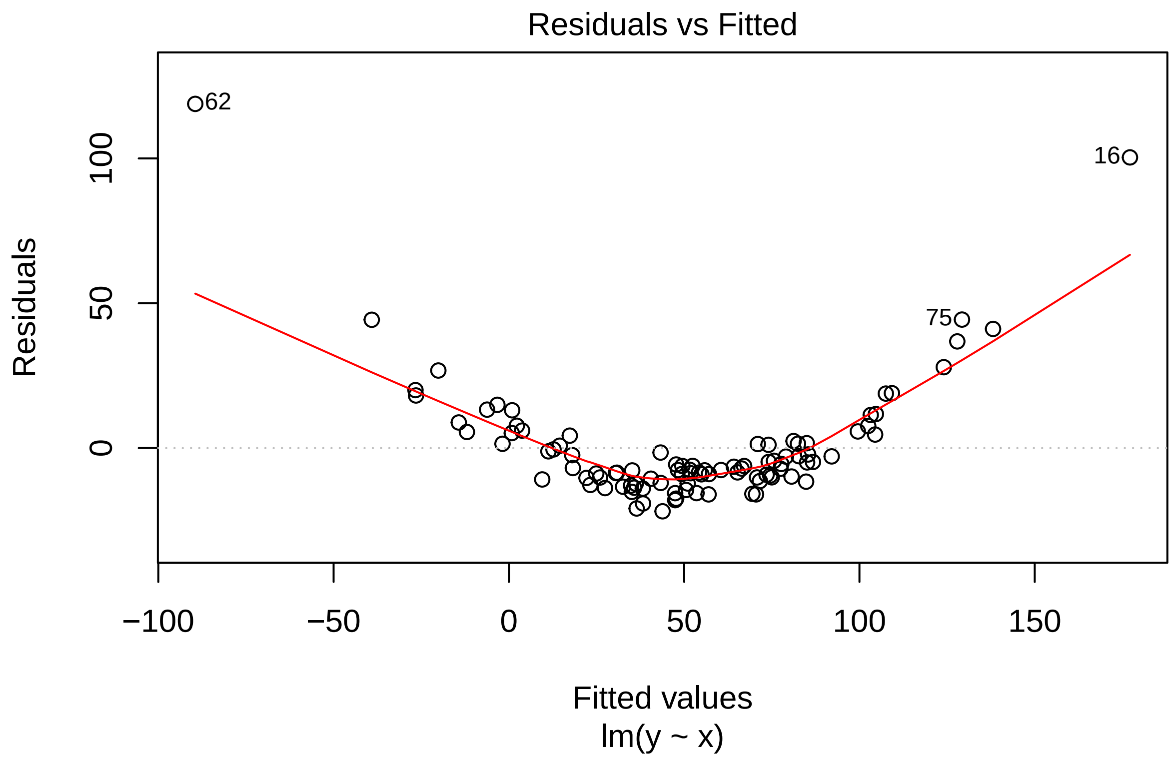
Scale−Location
31
20
43
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
−300
−200
−100
0
100
200
300
Fitted values
lm(y ~ x)

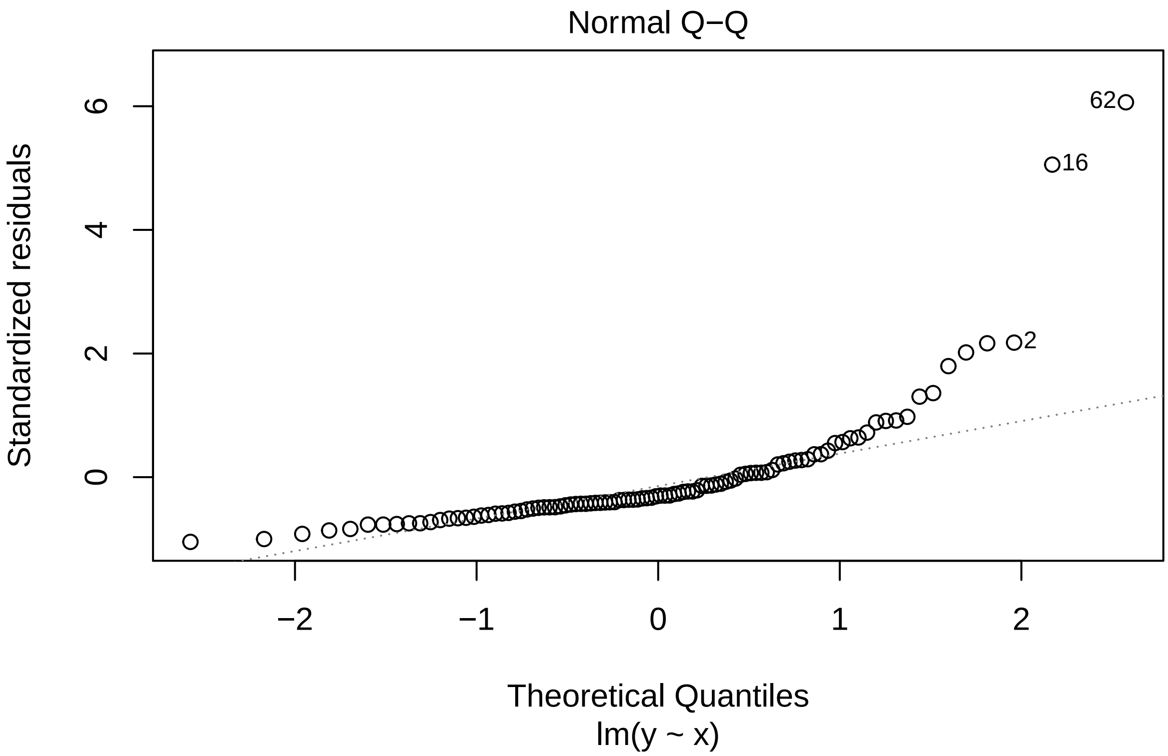


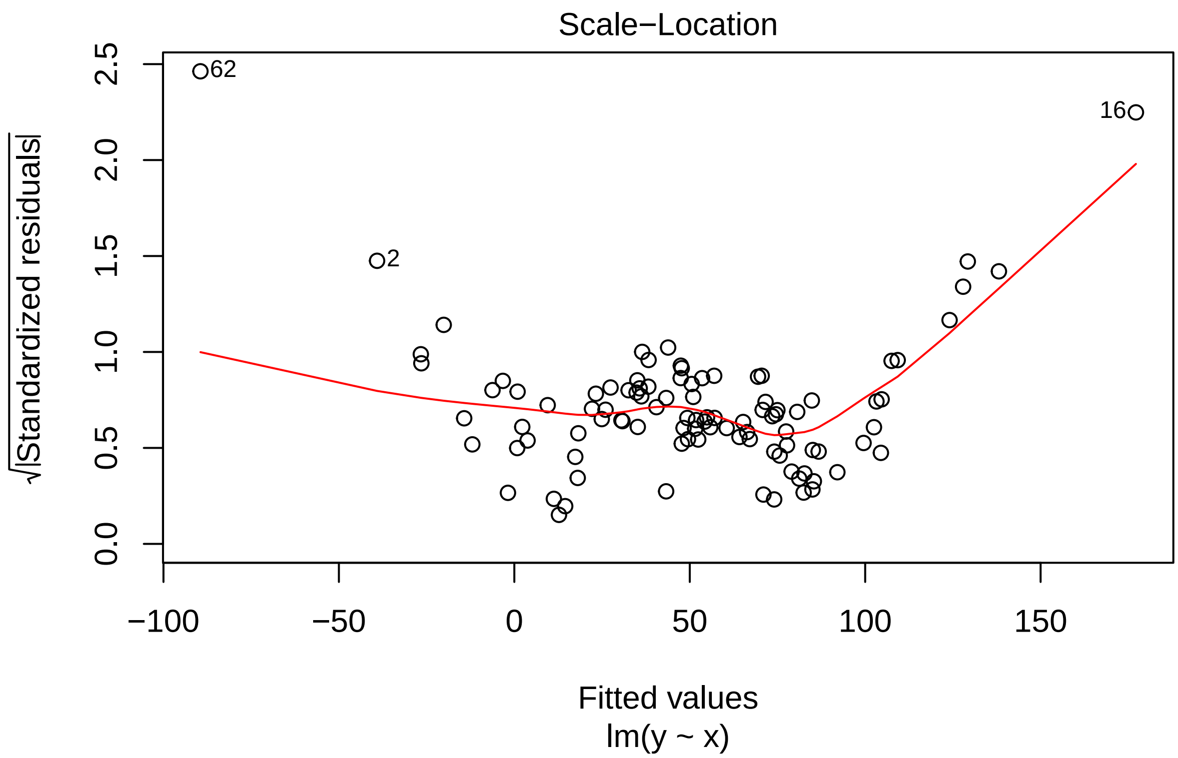

```r
# Missing squared term
eps <- rnorm(n=100, mean=0, sd= 1)
y <- 2 + 3*x + x**2  + 5*eps
plot(x,y)
abline(a=2, b=3, col="blue")
reg <- lm(y~x)
beta <- coef(reg)
names(beta) <- NULL
abline(a=beta[1], b=beta[2], col="green")
```



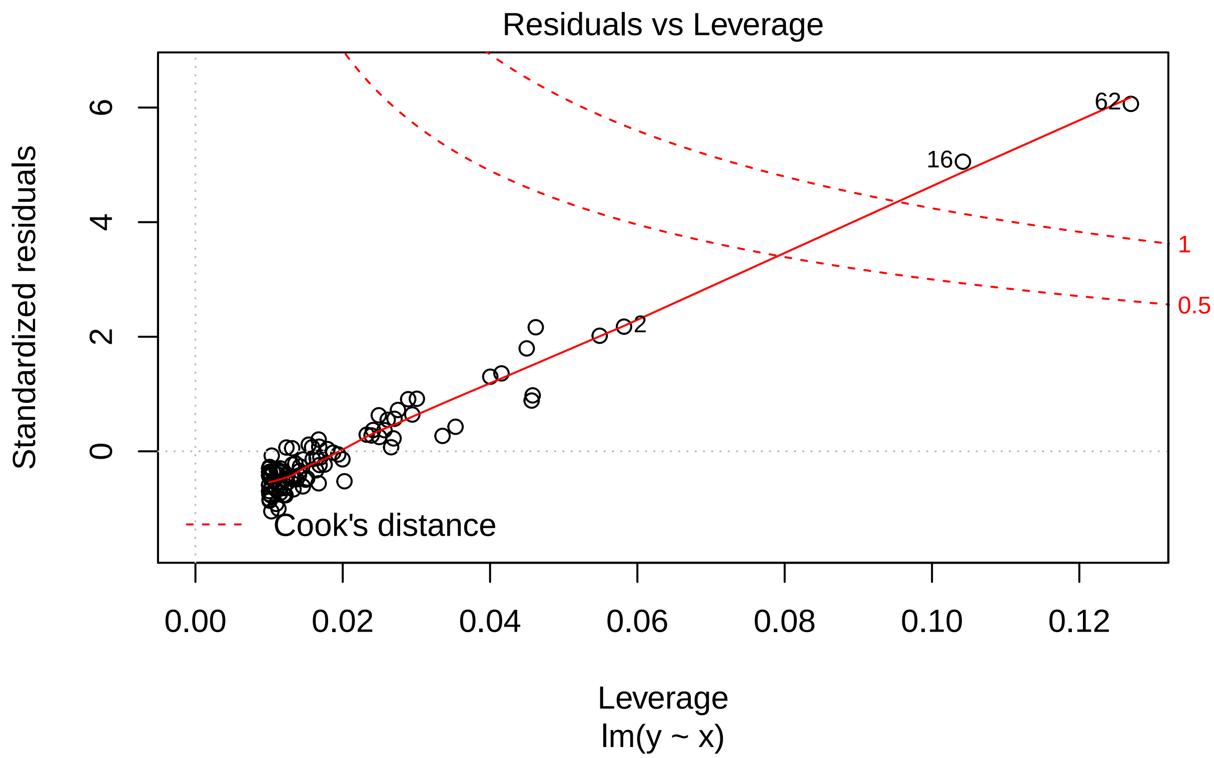

```r
plot(reg)
```


Residuals vs Fitted
Residuals
100
50
0
62
16
75
−100
−50
0
50
100
150
Fitted values
lm(y ~ x)


Normal Q−Q
Standardized residuals
0
2
4
6
62
16
2
−2
−1
0
1
2
Theoretical Quantiles
lm(y ~ x)



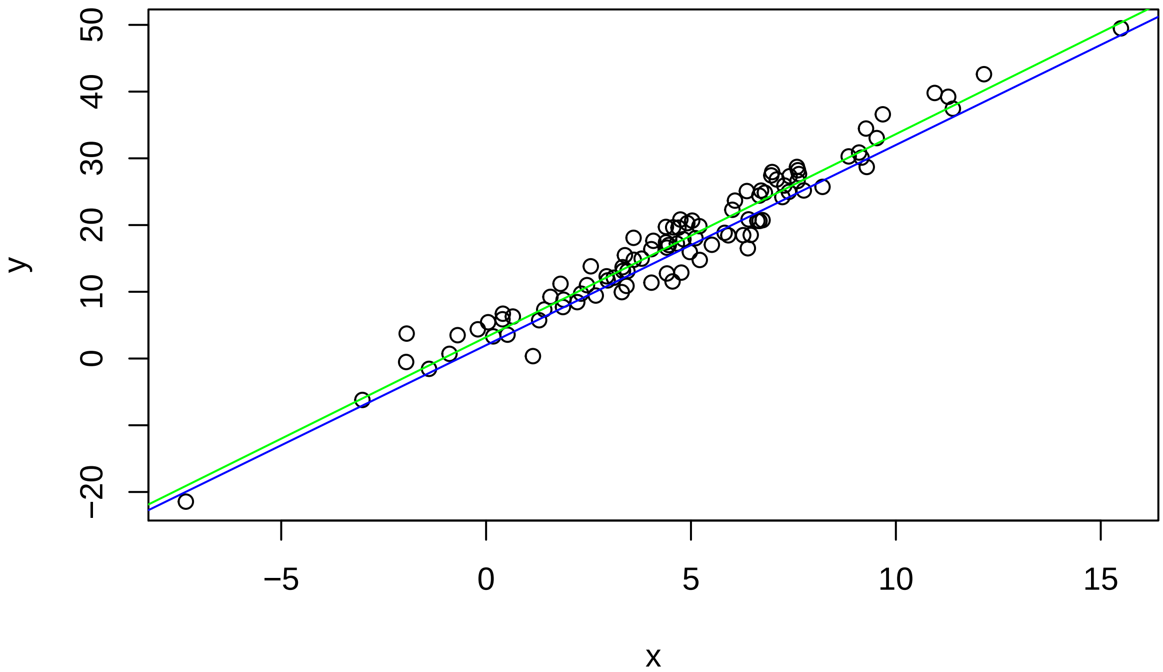
Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
62
2
16
−100
−50
0
50
100
150
Fitted values
lm(y ~ x)



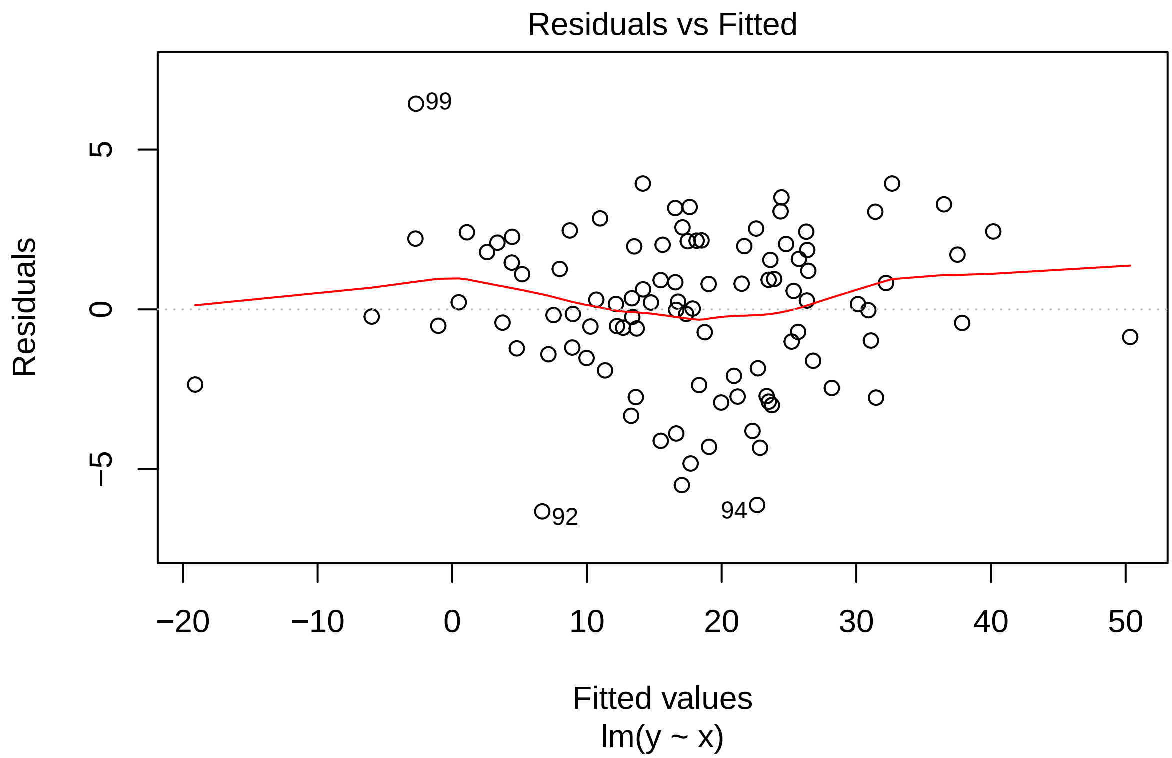

```
# Correlated errors
require(MASS)
```

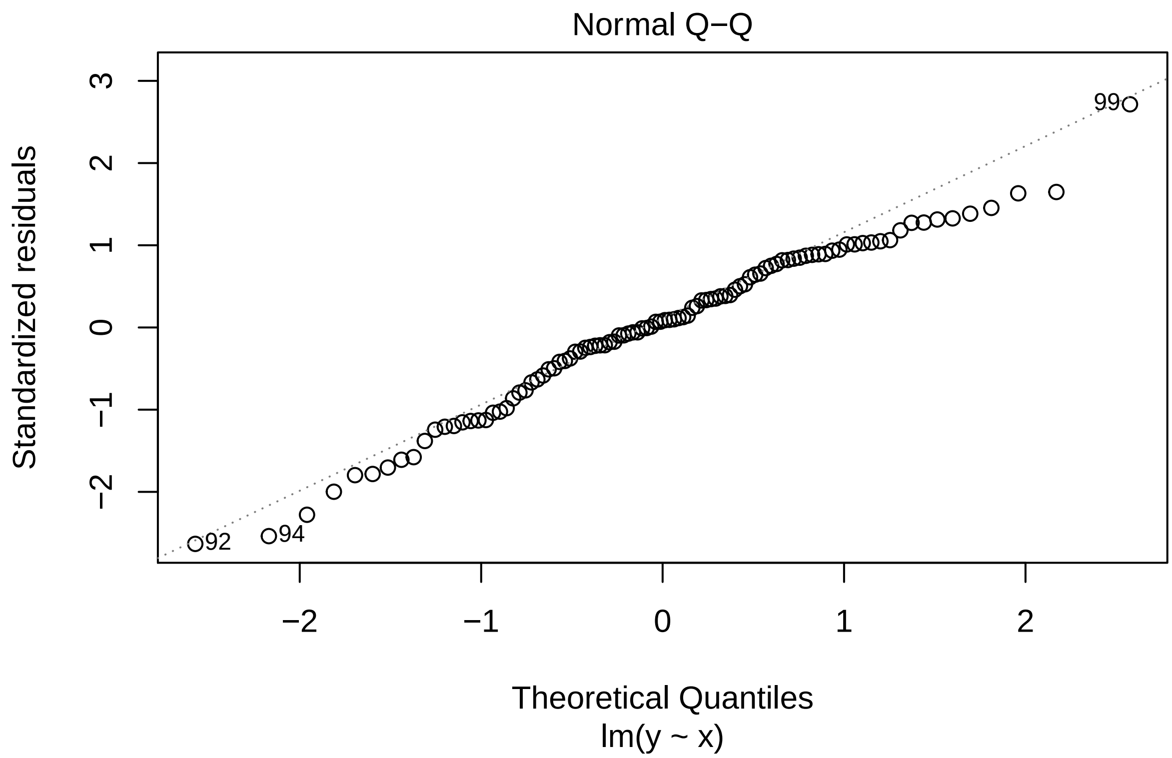
```
## Loading required package: MASS
```

```
Sigma <- matrix(0.7,100,100)
diag(Sigma) <- 1
eps <- mvrnorm(n = 1, mu = rep(0, length(x)), Sigma = Sigma)
y <- 2 + 3*x + 5*eps
plot(x,y)
abline(a=2, b=3, col="blue")
reg <- lm(y~x)
beta <- coef(reg)
names(beta) <- NULL
abline(a=beta[1], b=beta[2], col="green")
```

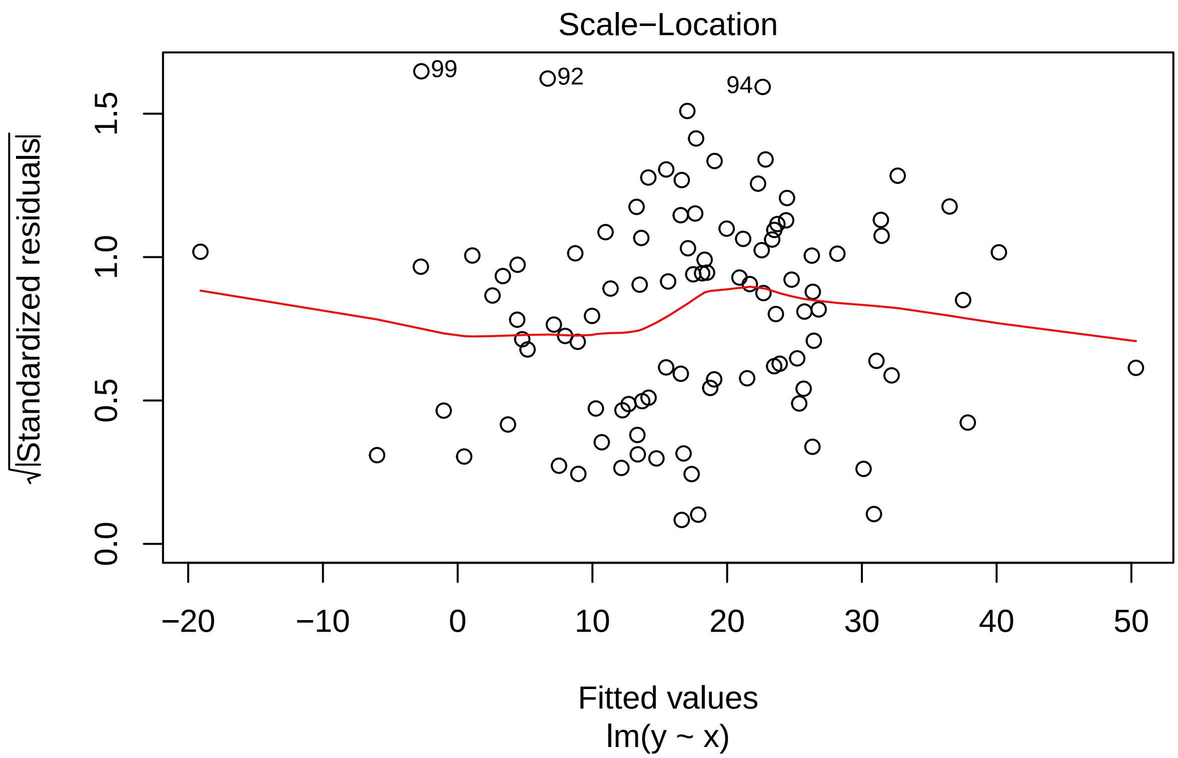



```r
plot(reg)
```

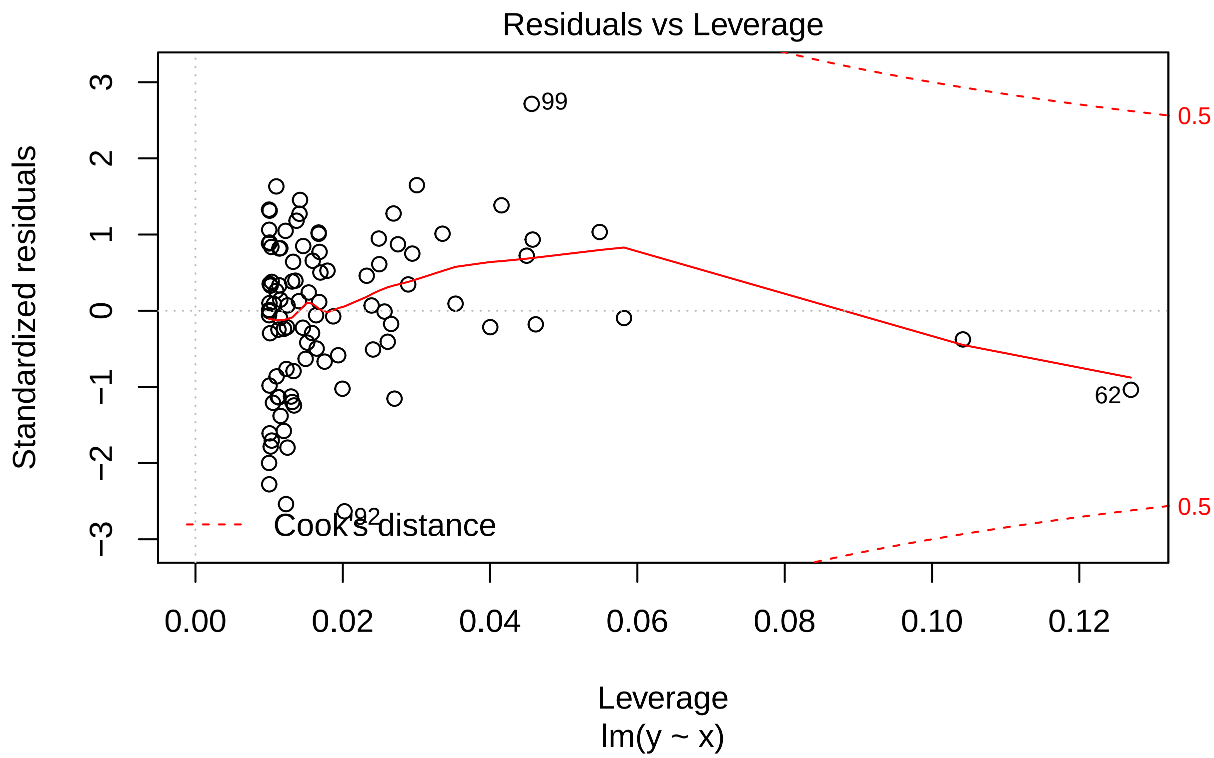

Residuals vs Fitted
Residuals
5
0
−5
99
92
94
−20
−10
0
10
20
30
40
50
Fitted values
lm(y ~ x)



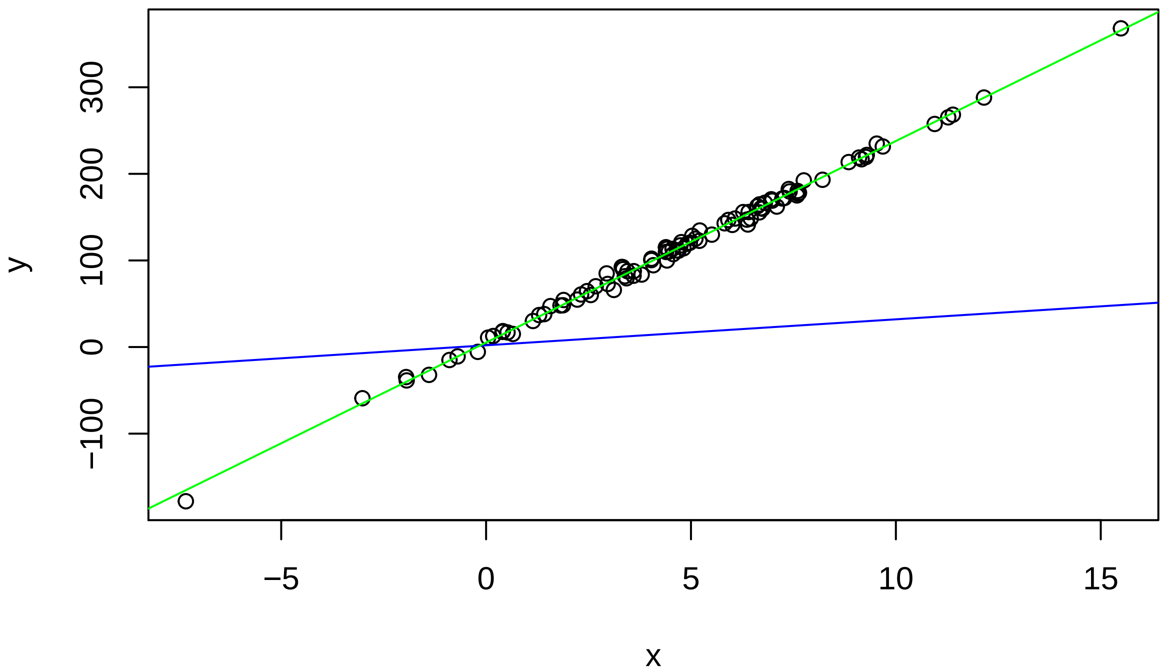
Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(y ~ x)
3
2
1
0
−1
−2
−2
−1
0
1
2
99
92
94



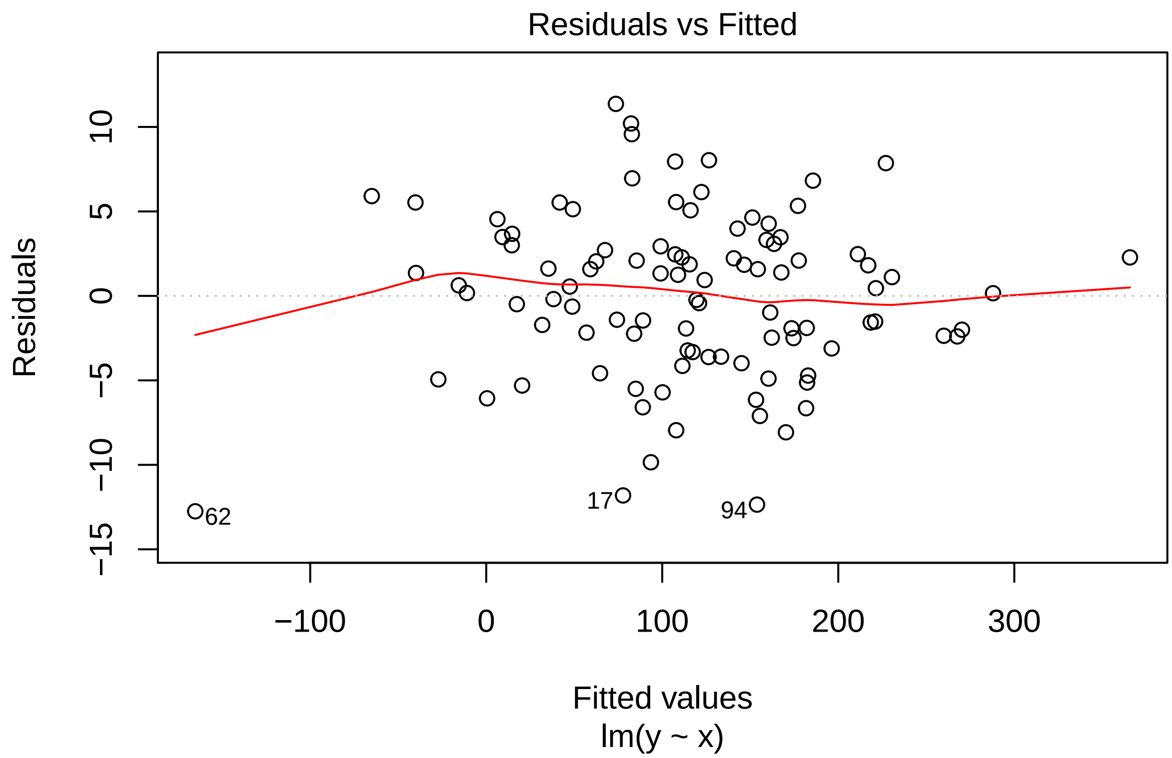
Scale−Location
99
92
94
√|Standardized residuals|
0.0
0.5
1.0
1.5
−20
−10
0
10
20
30
40
50
Fitted values
lm(y ~ x)

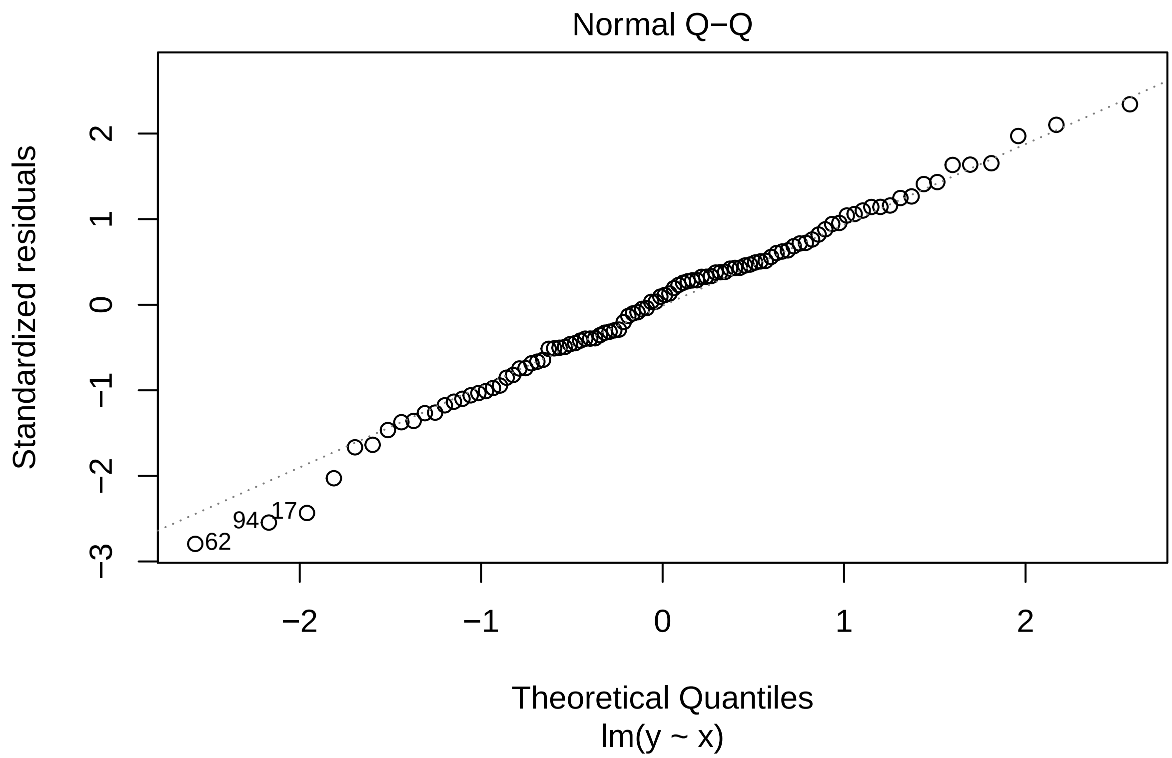


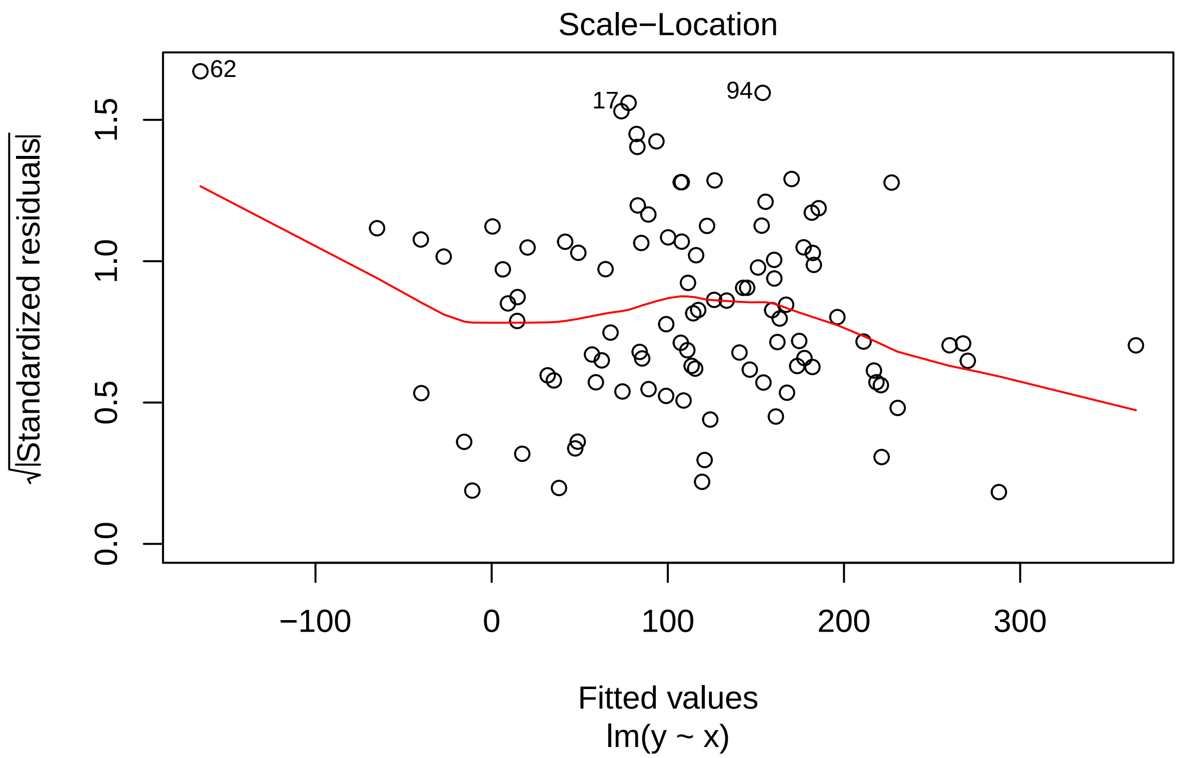

```r
# x, epsilon dependent
eps <- rnorm(n=100, mean=x*4+1, sd= 1)
y <- 2 + 3*x + 5*eps
plot(x,y)
abline(a=2, b=3, col="blue")
reg <- lm(y~x)
beta <- coef(reg)
names(beta) <- NULL
abline(a=beta[1], b=beta[2], col="green")
```



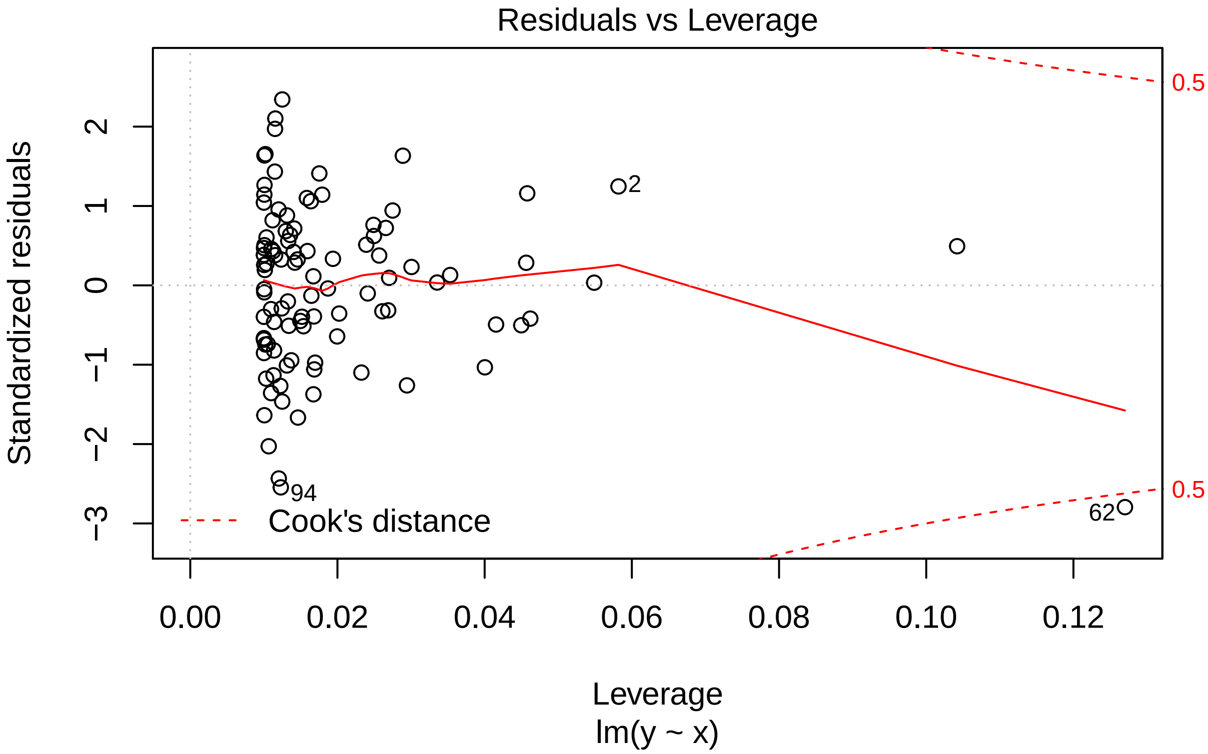

```
plot(reg)
```


Residuals vs Fitted
Residuals
10
5
0
−5
−10
−15
−100
0
100
200
300
Fitted values
lm(y ~ x)
62
17
94


Normal Q−Q
Standardized residuals
−3
−2
−1
0
1
2
94
17
62
−2
−1
0
1
2
Theoretical Quantiles
lm(y ~ x)



Scale−Location
62
17
94
√|Standardized residuals|
0.0
0.5
1.0
1.5
−100
0
100
200
300
Fitted values
lm(y ~ x)



Residuals vs Leverage
Standardized residuals
Leverage
lm(y ~ x)
Cook's distance
0.5
0.5
2
94
62
−3
−2
−1
0
1
2
0.00
0.02
0.04
0.06
0.08
0.10
0.12