Simulation of Regression Models

COSC/DATA 405/505

Simulating Data from Regression Models

The simple linear regression *model* relating a *response variable* y to a *predictor variable* x is

$$y = \beta_0 + \beta_1 x + \varepsilon$$

where β_0 is the intercept and β_1 is the slope of the regression line.

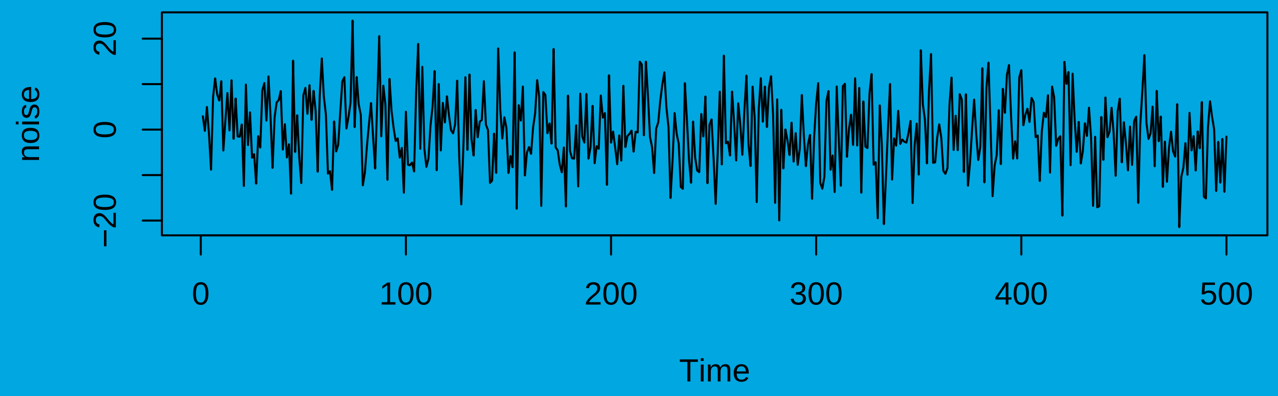
ε is a random quantity representing noise about the line.

The noise is often assumed to be a sequence of independent normal random variables with mean 0 and constant variance σ^2.

e.g. consider 500 values of ε which have $\sigma = 8$:

```
eps <- rnorm(500, sd = 8)
```

```
ts.plot(eps, ylab="noise")
```



Simulating Regression Data

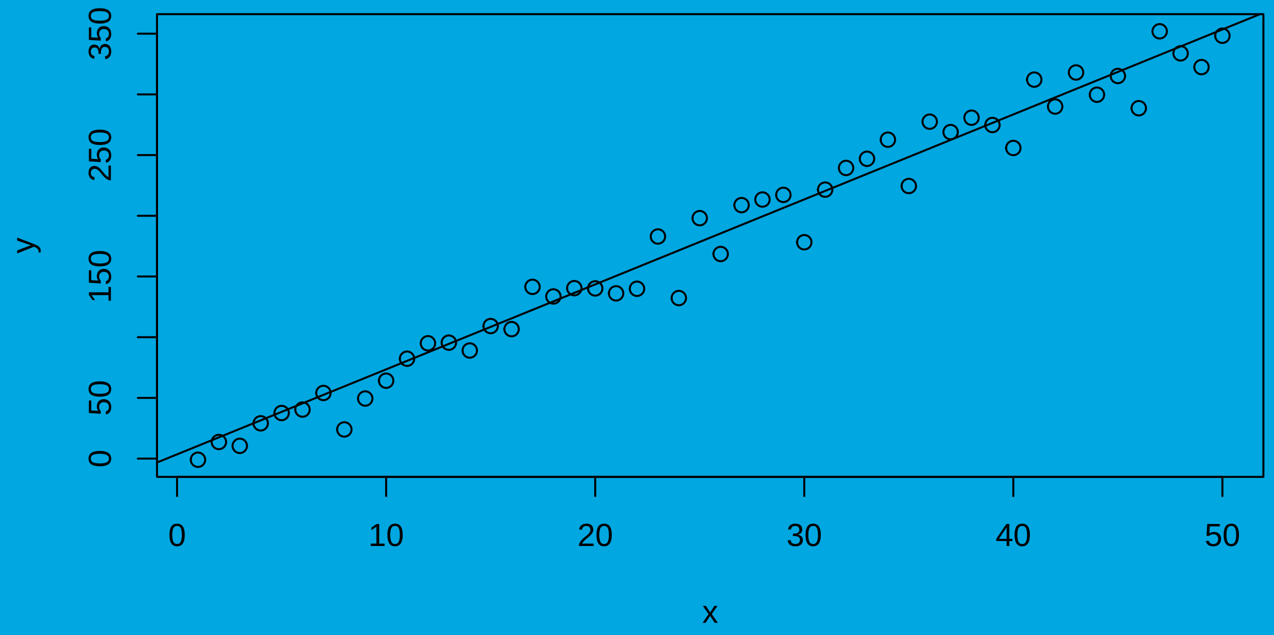
In the simple linear regression model, the noise is added to a line of slope β_1 and intercept β_0.

e.g. Suppose x values are taken at $\{1, 2, 3, \ldots, 50\}$. If the slope is 3.5 and the intercept is 7.0, and the noise is normal with standard deviation 16.0, we have

```
x <- 1:50
eps <- rnorm(50, sd = 16)
y <- 3.5 + 7.0*x + eps
```

Simulating Regression Data

```
plot(y ~ x)
abline(3.5, 7)
```



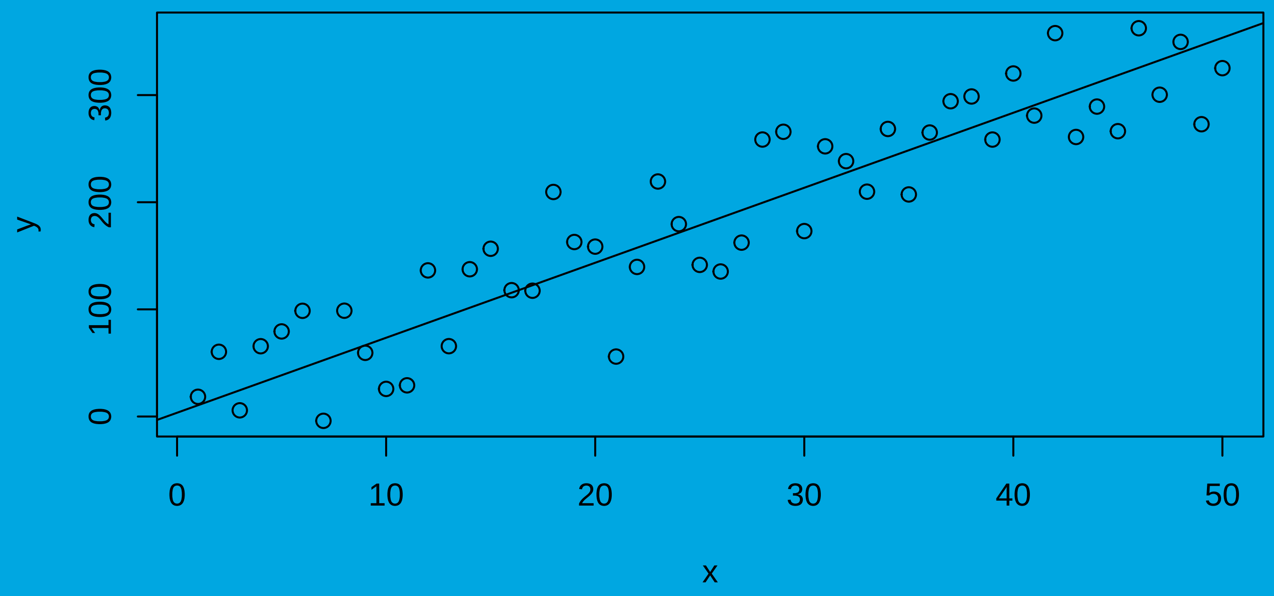
Simulating Regression Data

Suppose the standard deviation is larger: 40.0, we have

```
x <- 1:50
eps <- rnorm(50, sd = 40 )
y <- 3.5 + 7.0*x + eps
```

Simulating Regression Data

```
plot(y ~ x)
abline(3.5, 7)
```



... larger noise standard deviation gives more variation about the true line ...

Regression Data Example

The `p2.12` data frame in the *MPV* package has 12 observations on the number of pounds of steam used per month at a plant and the average monthly ambient temperature.

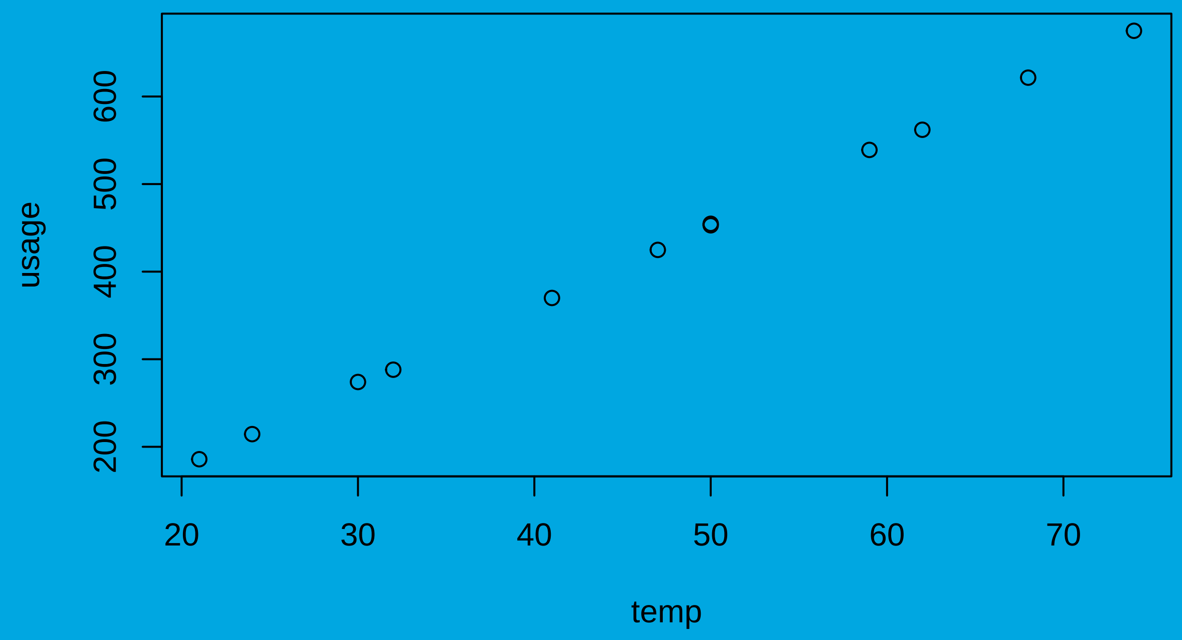
This data frame contains the following columns:

`temp` **ambient temperature (in degrees F)**

`usage` **usage (in thousands of pounds)**

Plotting the Data

```r
library(MPV)
plot(usage ~ temp, data = p2.12)
```



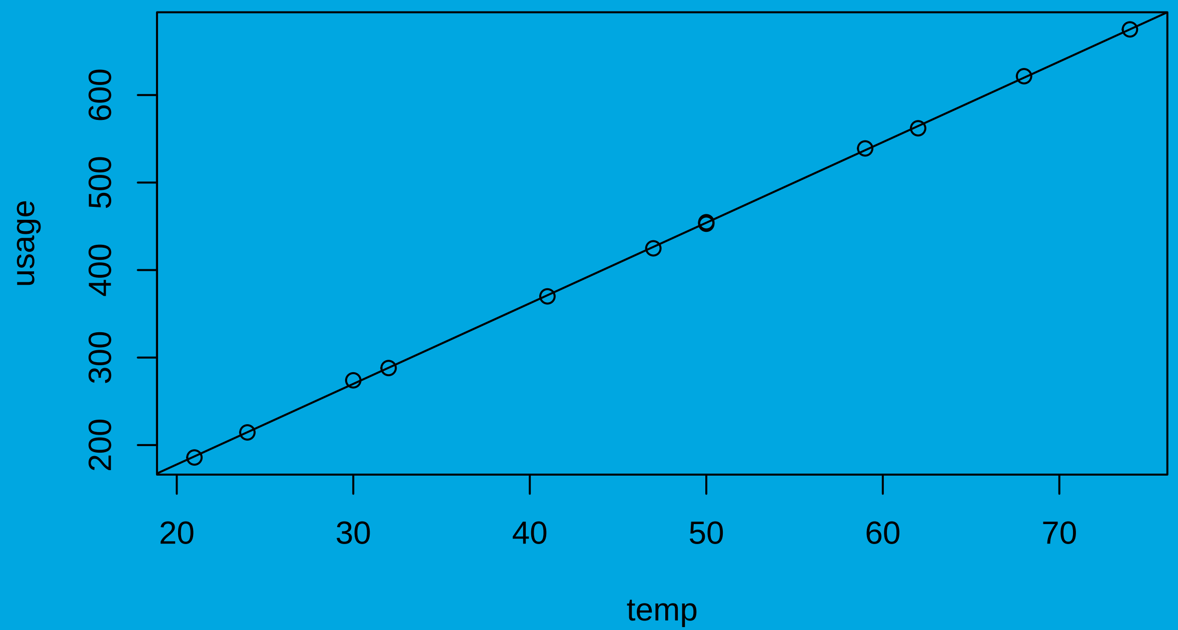
```
usage.lm <- lm(usage ~ temp, data = p2.12)
summary(usage.lm)$coefficients

##              Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) -6.332087 1.67004573  -3.791565 3.534310e-03
## temp         9.208468 0.03382295 272.254999 1.099192e-20
```

Fitted line: $\widehat{y} = -6.332 + 9.208x$

The p-values given in the right hand column are both small $\rightsquigarrow$ strong evidence that the intercept and slope are nonzero. We will see what this means, using simulation, later on in this set of slides.

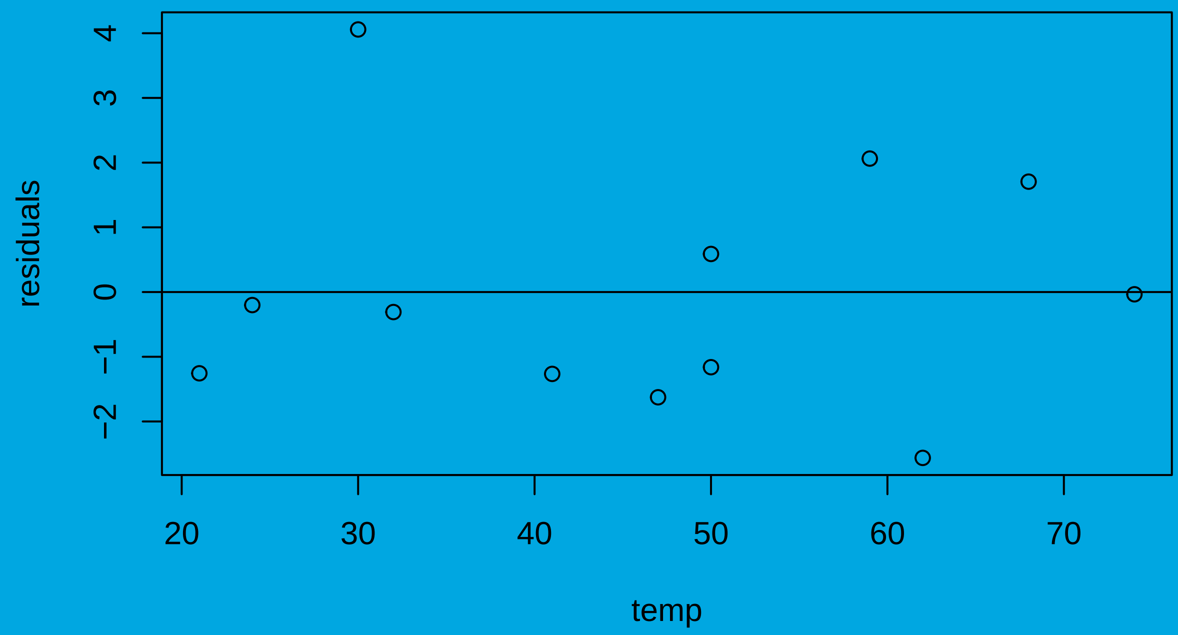
Plotting the Best Fit Line

```
plot(usage ~ temp, data = p2.12)
abline(usage.lm)
```



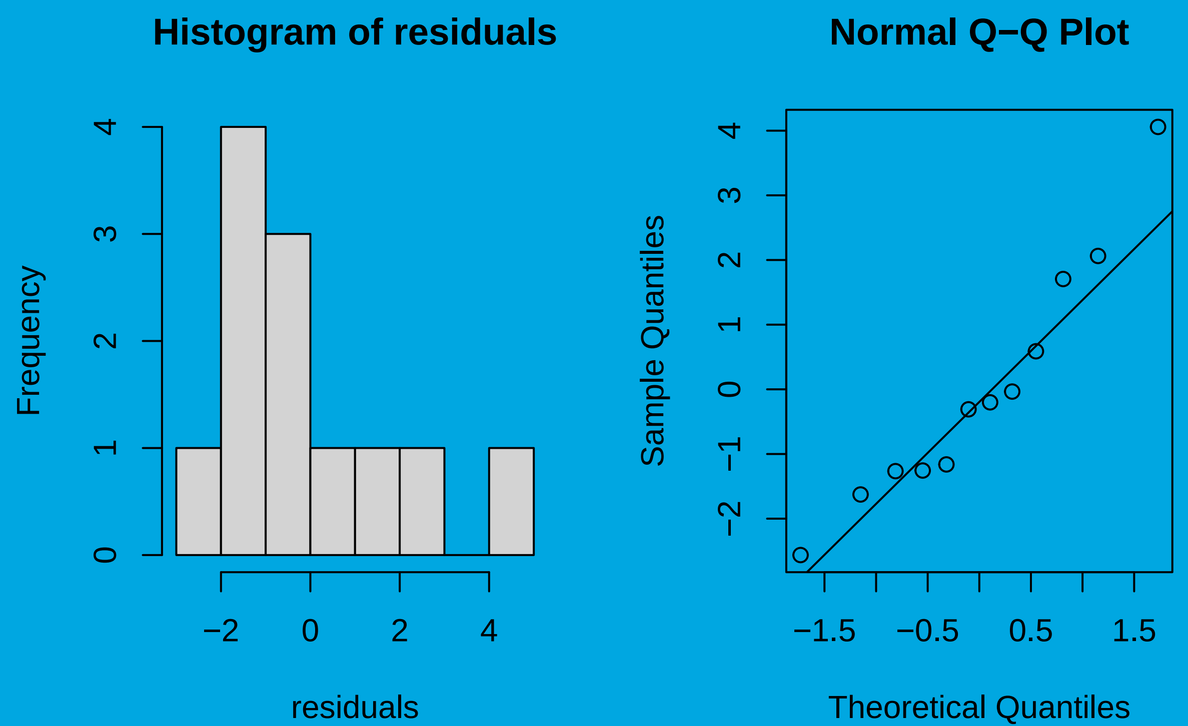
Residuals: Estimates of the Error

```
residuals <- resid(usage.lm)
plot(residuals ~ temp, data = p2.12)
abline(h = 0)
```



Modelling the Error

```
par(mfrow=c(1,2))
hist(residuals)
qqnorm(residuals); qqline(residuals)
```



Normality is reasonable. Mean is 0 but what is the standard deviation?

Estimating the noise standard deviation

```
summary(usage.lm)$sigma

## [1] 1.945628
```

The regression procedure is based on mathematics which would take too long to go through here – there are other courses that cover that material.

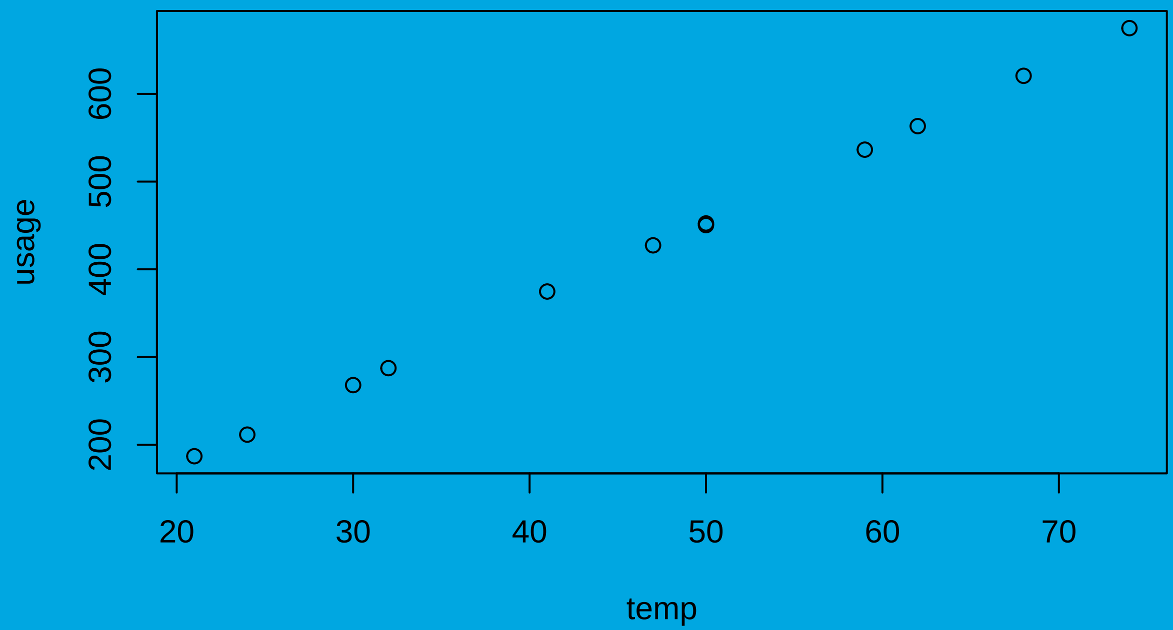
Instead, we can use simulation to gain intuition into the procedure.

By simulating new data where we know the true coefficients and the true errors, we can see how the regression estimates differ from the truth.

We can also learn some things about the residuals and how they relate to the true errors.

We will simulate data that "looks" like the data in `p2.12`:

```
p2.12sim <- p2.12  # p2.12sim will soon contain simulated data
eps <- rnorm(n = nrow(p2.12sim) , sd = 1.945) # simulated noise
p2.12sim$usage <- -6.332 + 9.208*p2.12sim$temp +eps
plot(usage ~ temp, data = p2.12sim)
```



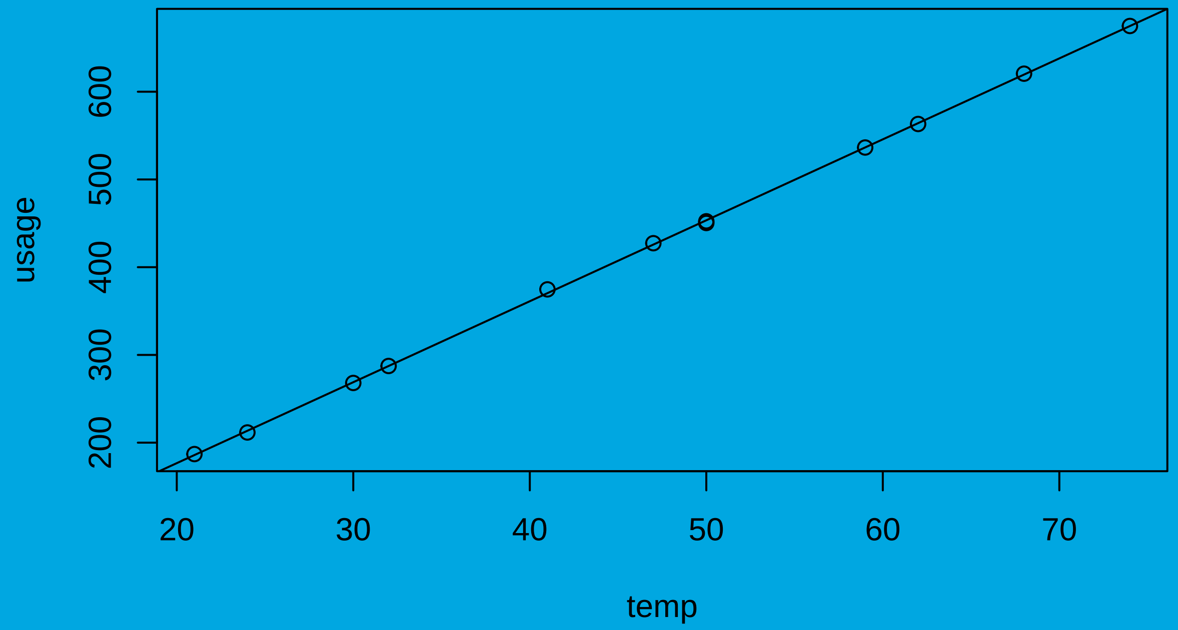
Simulated Linear Regression Data

```
p2.12sim.lm <- lm(usage ~ temp, data = p2.12sim)
#estimated intercept and slope for simulated data
coef(p2.12sim.lm)

## (Intercept)        temp
##   -7.905550    9.227047

summary(p2.12sim.lm)$sigma  # sd estimate

## [1] 1.988767
```

Now we see the estimates of the intercept, slope and estimate of σ differ from the true values.

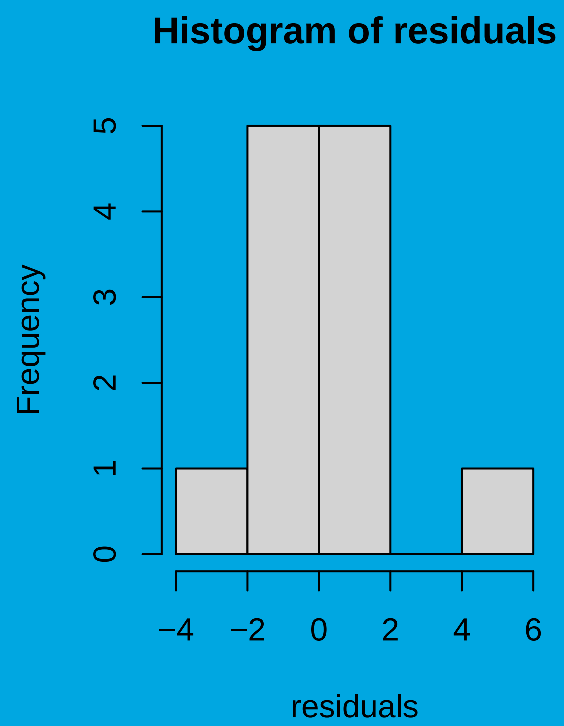
Plotting the Best Fit Line - Simulated Data

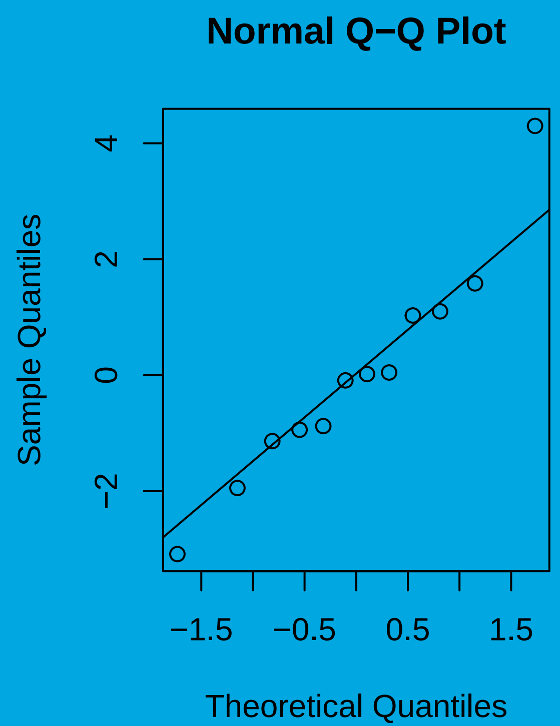
```
plot(usage ~ temp, data = p2.12sim)
abline(p2.12sim.lm)
```



The True Errors are Normal; What about the Residuals?

```
residuals <- resid(p2.12sim.lm)
par(mfrow=c(1,2)); hist(residuals)
qqnorm(residuals); qqline(residuals)
```



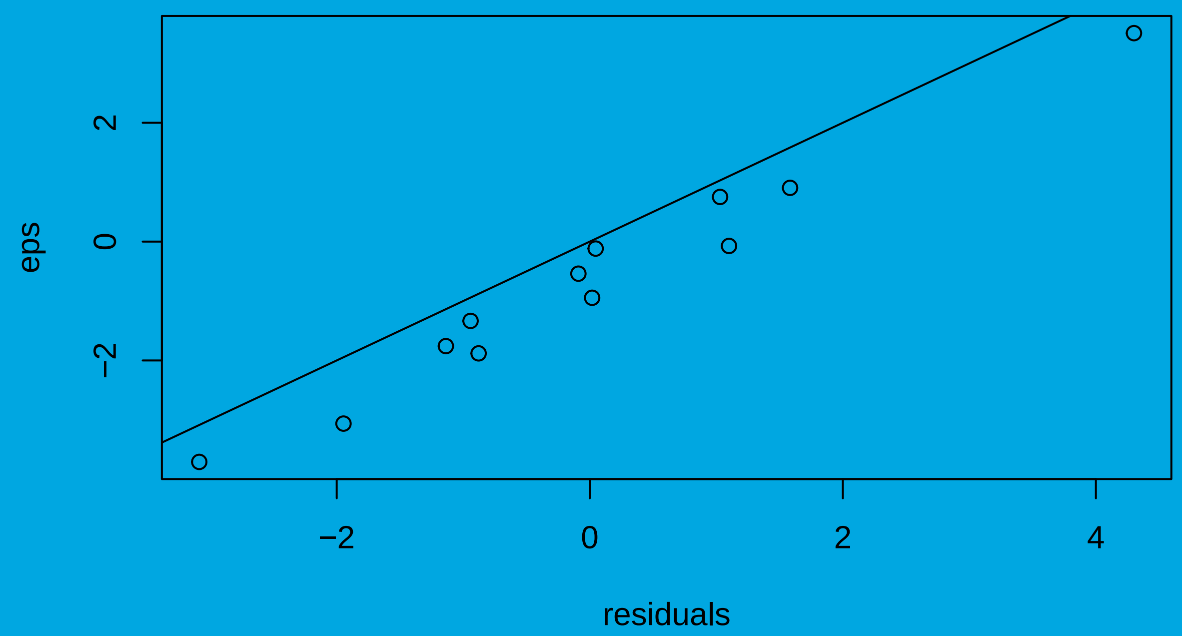


`qreference()` **could also be checked.**

How do the Simulated Residuals Behave?

Compare the simulated residuals with the true errors:

```
plot(eps ~ residuals)
abline(0,1)
```



By repeatedly simulating data sets and estimating the slope each time, we can see where some of the regression output comes from:

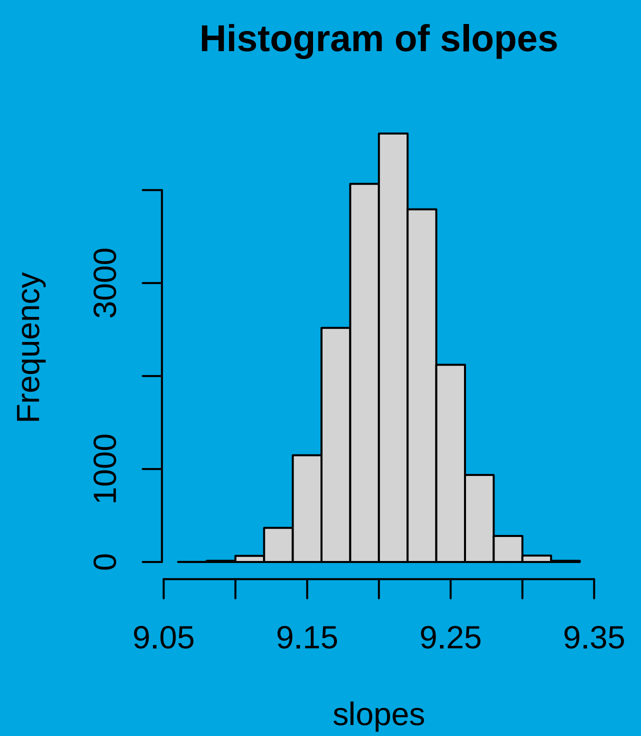
```
Nsims <- 20000; slopes <- sderrors <- numeric(Nsims)
for (i in 1:Nsims) {# 20000 simulated data sets
    eps <- rnorm(n = nrow(p2.12sim) , sd = 1.945)
    p2.12sim$usage <- -6.332 + 9.208*p2.12sim$temp +eps
    p2.12sim.lm <- lm(usage ~ temp, data = p2.12sim)
    slopes[i] <- coef(p2.12sim.lm)[2]
    sderrors[i] <- summary(p2.12sim.lm)$coefficients[2,2]
}
mean(slopes); sd(slopes)

## [1] 9.207975
## [1] 0.03406831
```

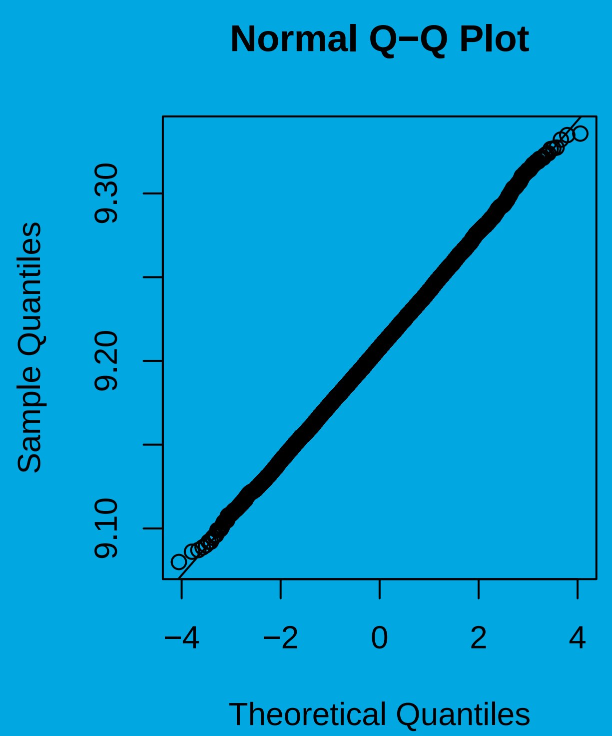
Compare with the estimate (9.208) and standard error (.0338) given on slide 9.

Note that we could do the same procedure for the intercept using coef()[1].

What is the distribution of the slope estimate?

```
par(mfrow=c(1,2)); hist(slopes); qqnorm(slopes); qqline(slope
```





... pretty convincing evidence that the slope estimate is approximately normally distributed ...

Testing whether the true slope is 0

This is summarized in the 3rd and 4th columns of the regression output on slide 4. What does it mean?

Statistical theory says that if the true slope is β, then the distribution of the

$$\frac{\textbf{slope estimate} - \beta}{\textbf{standard error estimate}}$$

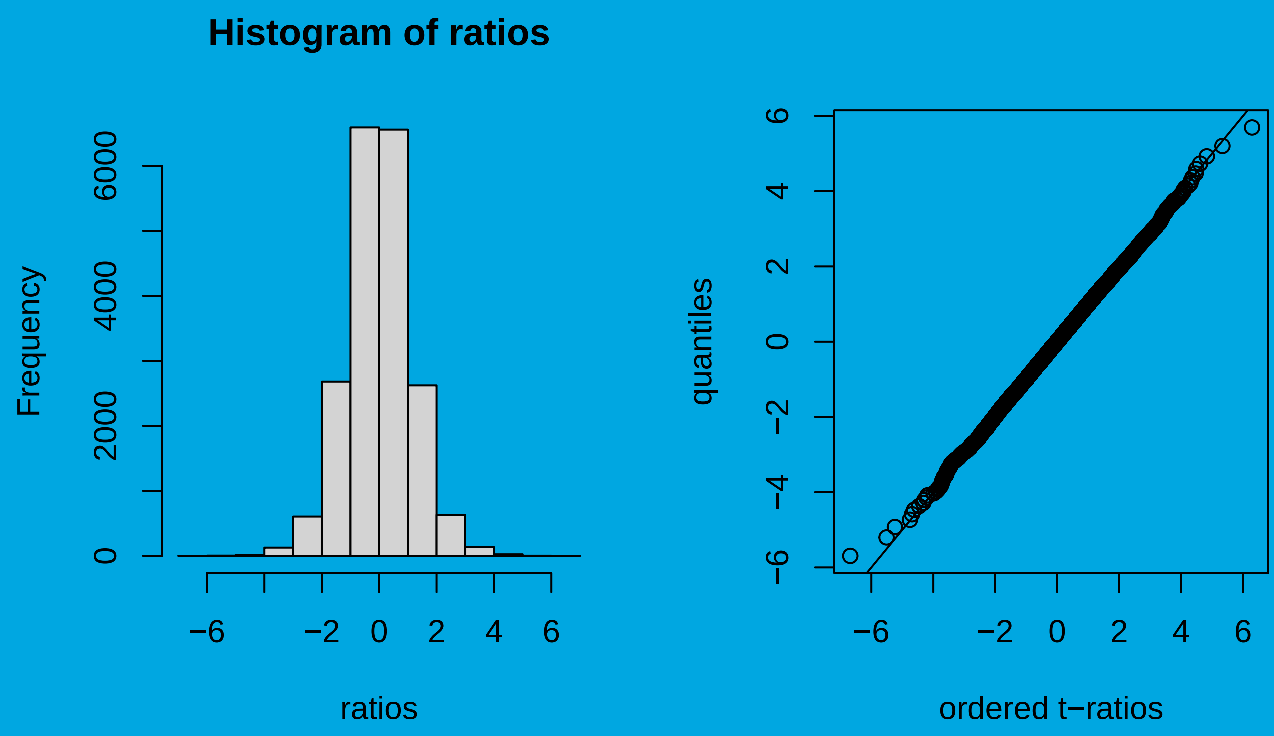
is a t distribution on $n - 2$ degrees of freedom.

We can verify this for the case where the true slope is 9.208 and $n = 12$:

```
ratios <- (slopes - 9.208)/sderrors
n <- nrow(p2.12)
```

Plotting the t ratios

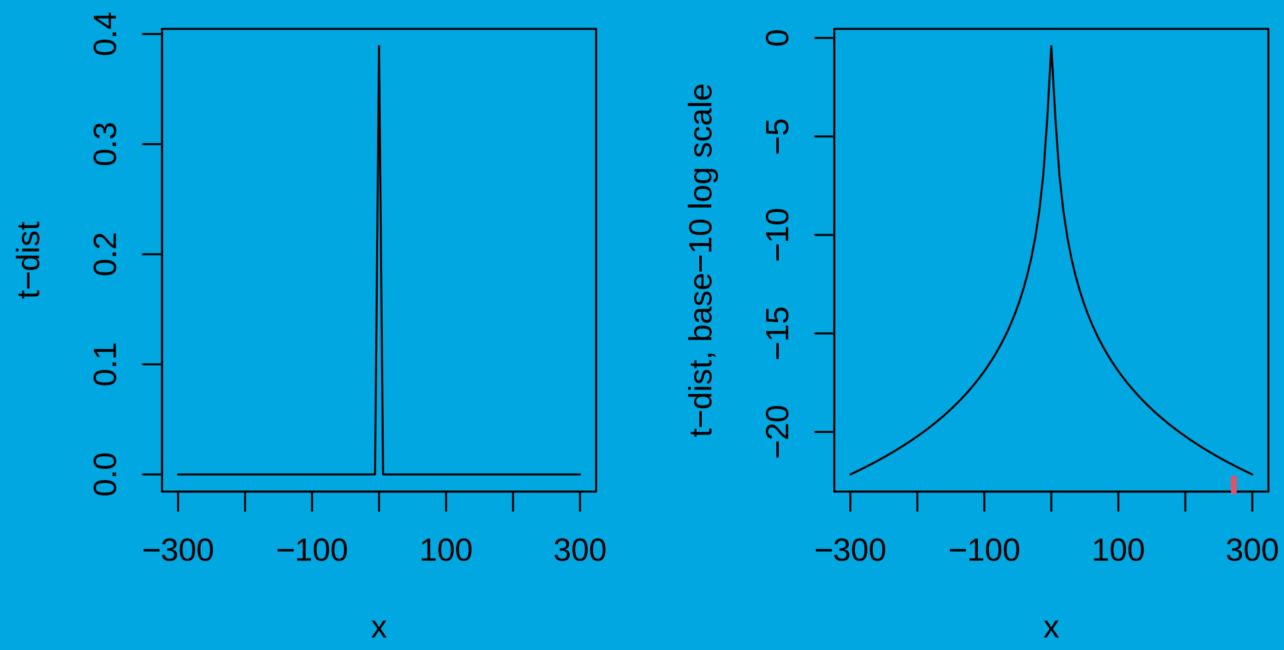
```
par(mfrow=c(1,2)); hist(ratios)
qqplot(ratios, qt((1:9999)/10000, df = n-2),
    ylab="quantiles", xlab="ordered t-ratios")
abline(0,1)
```



... pretty convincing evidence that the t-ratios follow a t-distribution on 10 degrees of freedom ...

How believable is it that the true slope is 0?

```
obstRatio <- 9.208/.0338 # observed t-ratio
par(mfrow=c(1,2))
curve(dt(x, df=10), -300, 300, ylab="t-dist")
curve(log(dt(x, df=10), base=10), -300, 300,
    ylab="t-dist, base-10 log scale")
rug(obstRatio, col=2, lwd=3) # locate the observed t-ratio
```



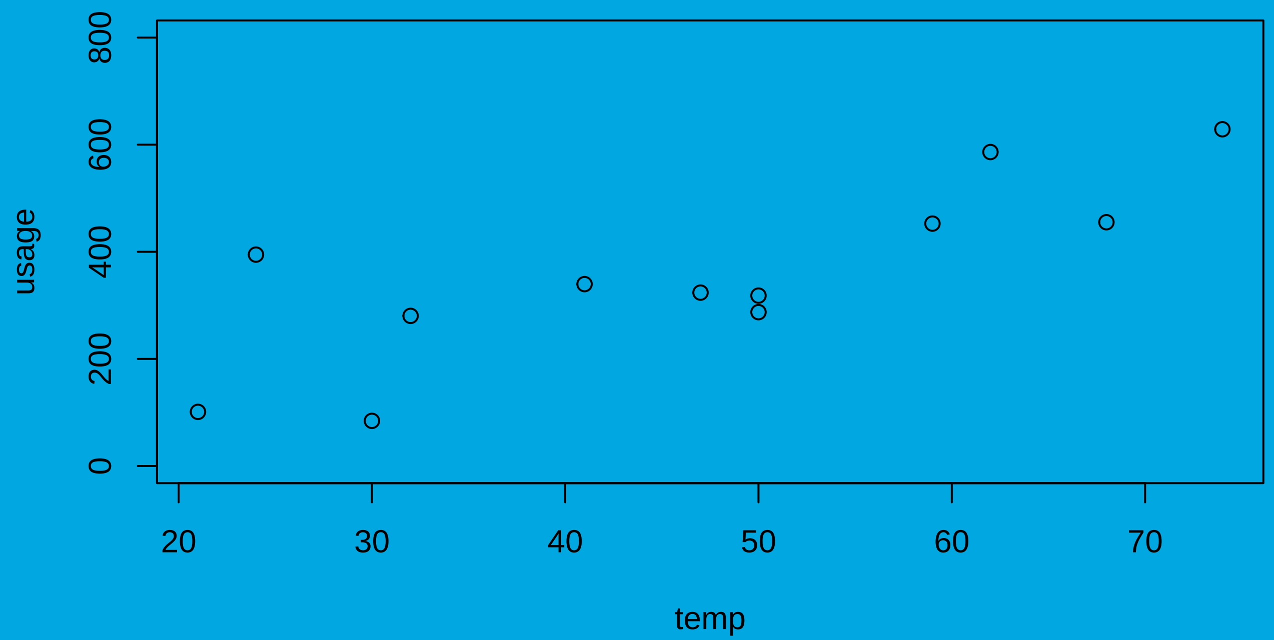
The probabilities are so tiny that we use a base-10 log to visualize the probability that we could see such a large slope if the true slope were 0.

The probability, which is the area under the curve to the right of 272.25, is less than 10^{-20} – this is the p-value for the slope given on slide 4.

Simulated Linear Regression Data - Noisier

We can use simulation to consider other scenarios. What if conditions change so that there is more variability?

```
eps <- rnorm(nrow(p2.12), sd = 100) # simulated noise - larger sd
p2.12sim$usage <- -6.332 + 9.208*p2.12sim$temp +eps
plot(usage ~ temp, data = p2.12sim, ylim = c(0, 800))
```



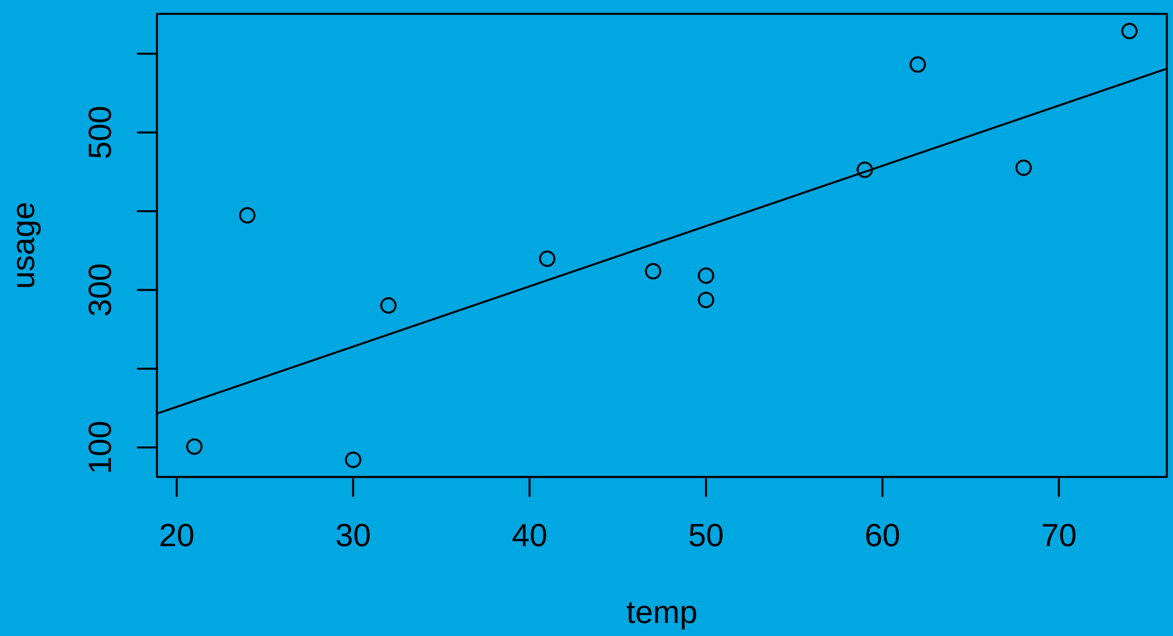
Simulated Linear Regression Data – Noisier

```
p2.12sim.lm <- lm(usage ~ temp, data = p2.12sim)
#estimated intercept and slope for simulated data
coef(p2.12sim.lm)

## (Intercept)        temp
##   -1.471046    7.652513

summary(p2.12sim.lm)$sigma  # sd estimate

## [1] 103.0971
```

The estimates of the intercept, slope and estimate of σ differ a lot from the true values.

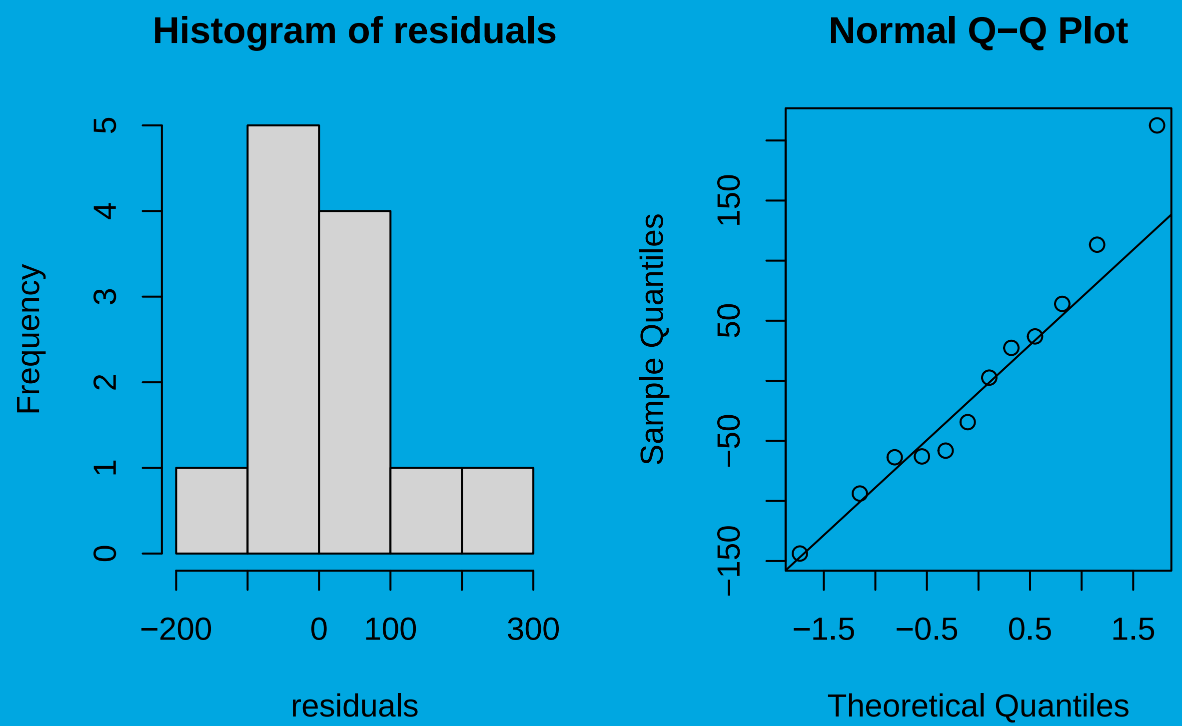
Plotting the Best Fit Line - Noisier Data

```
plot(usage ~ temp, data = p2.12sim)
abline(p2.12sim.lm)
```



How do the Noisier Simulated Residuals Behave?

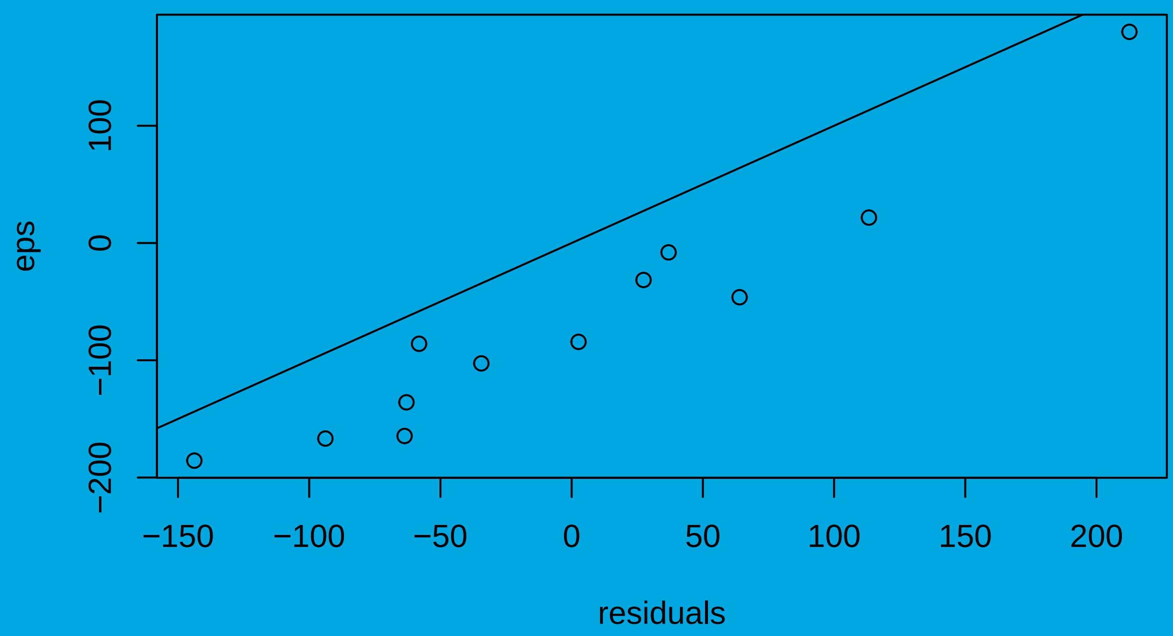
```r
residuals <- resid(p2.12sim.lm)
par(mfrow=c(1,2))
hist(residuals)
qqnorm(residuals); qqline(residuals)
```



How do the Noisier Simulated Residuals Behave?

Compare the simulated residuals with the true errors:

```
plot(eps ~ residuals)
abline(0,1)
```



Notice that the scale on the vertical axis is much larger than before. Why?

The residuals differ more from the true errors than before.

Another Example: Model Car Data

Consider the data on the model car that was released from various points on a ramp and the distance traveled was measured.

```
library(DAAG)
mcar.lm <- lm(distance.traveled ~ starting.point,
      data = modelcars)
summary(mcar.lm)$coefficients

##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)    8.083333  1.0779514  7.498792 2.065661e-05
## starting.point 2.013889  0.1312041 15.349288 2.801914e-08
```

The fitted model is

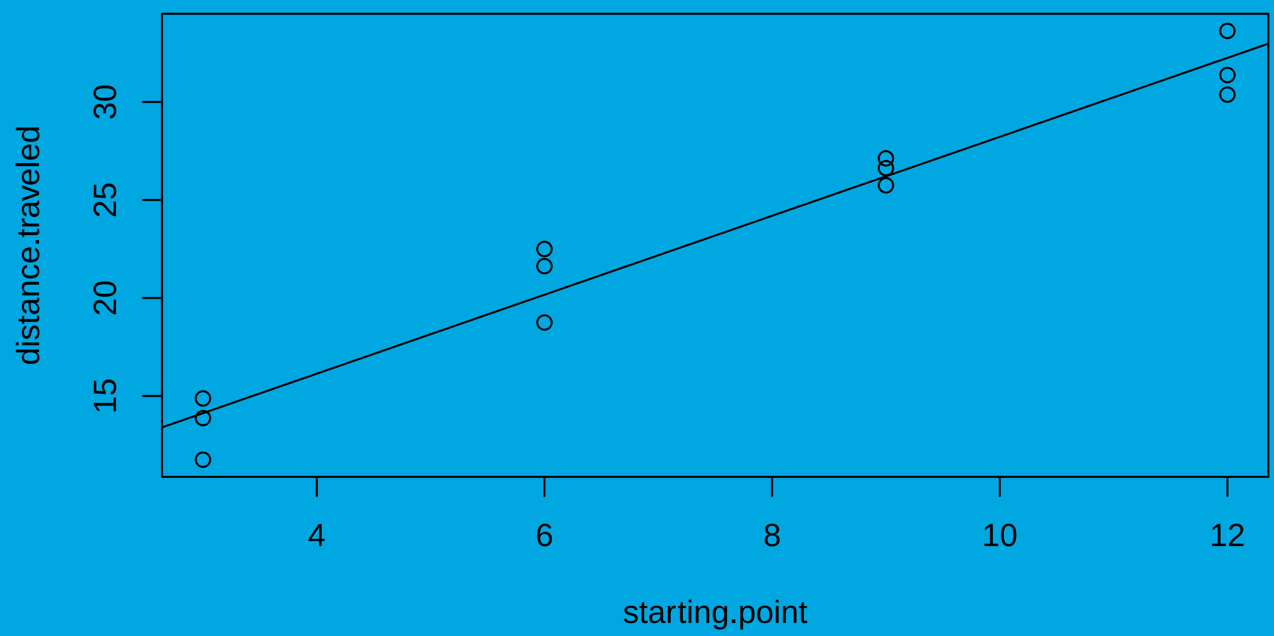
$$y = 8.0833333 + 2.0138889x + \varepsilon$$

where y is distance and x is starting point. The error (ε) standard deviation is

```
summary(mcar.lm)$sigma

## [1] 1.524453
```

Plotting the Model Car Data

```r
plot(distance.traveled ~ starting.point,
     data = modelcars)
abline(mcar.lm)
```



Use Simulation to Test the Slope

```
b0 <- coef(mcar.lm)[1]
b1 <- coef(mcar.lm)[2]
sdCar <- summary(mcar.lm)$sigma
```

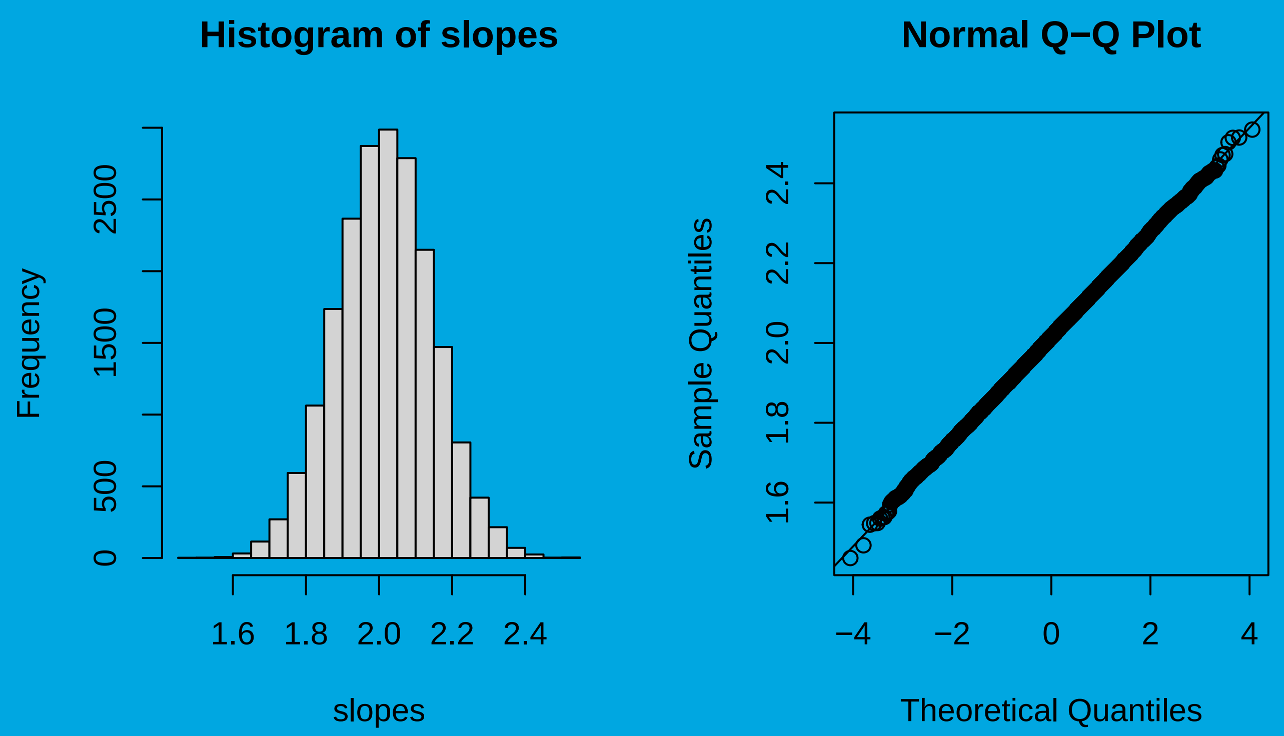
```
Nsims <- 20000; slopes <- sderrors <- numeric(Nsims)
for (i in 1:Nsims) {# 20000 simulated data sets
    eps <- rnorm(n = nrow(modelcars) , sd = sdCar)
    modelcars$distance.traveled <-
           b0 + b1*modelcars$starting.point +eps
    mcar.lm <- lm(distance.traveled ~ starting.point,
           data = modelcars); slopes[i] <- coef(mcar.lm)[2]
    sderrors[i] <- summary(mcar.lm)$coefficients[2,2]
}
mean(slopes); sd(slopes)
```

```
## [1] 2.015052
## [1] 0.1308729
```

Compare with the estimate (2.014) and standard error (.1312) given on slide 31.

What is the distribution of the slope estimate?

```r
par(mfrow=c(1,2)); hist(slopes); qqnorm(slopes); qqline(slopes)
```

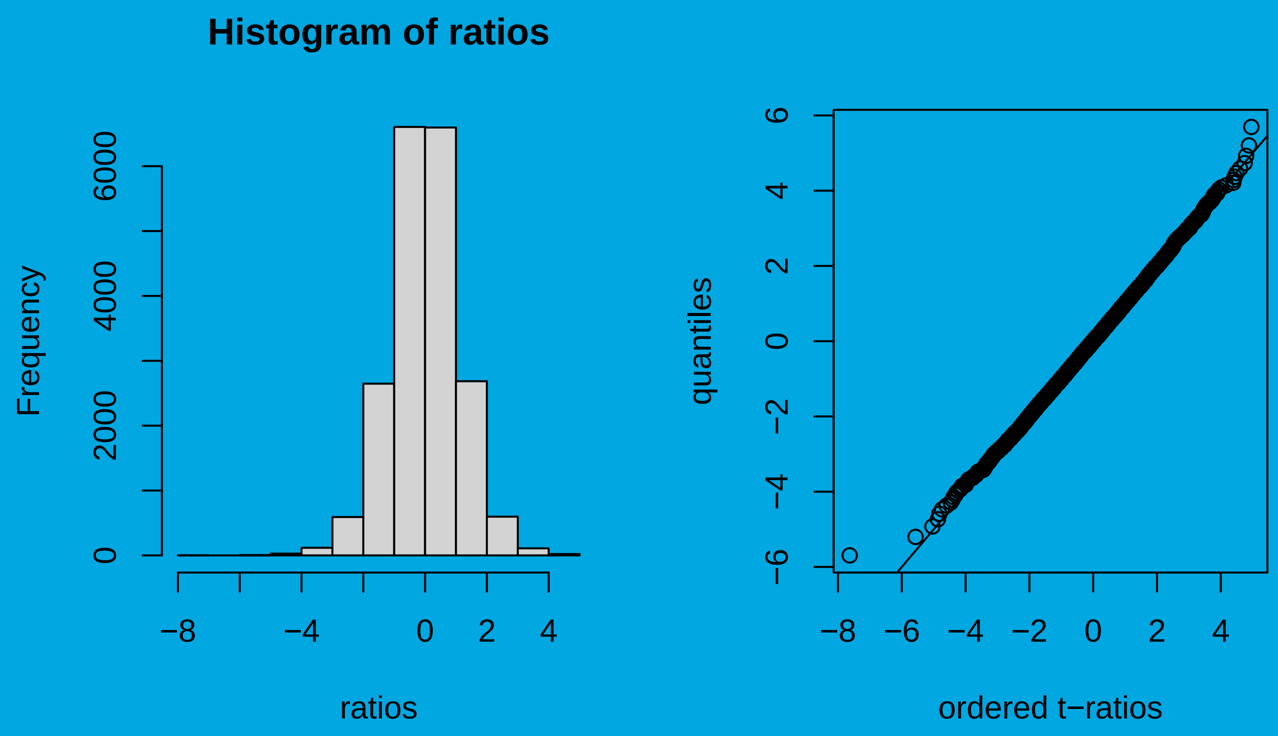


... additional evidence that the slope estimate is approximately normally distributed ...

Testing whether the true slope is 0

```
ratios <- (slopes - 2.015)/sderrors
n <- nrow(modelcars)
```

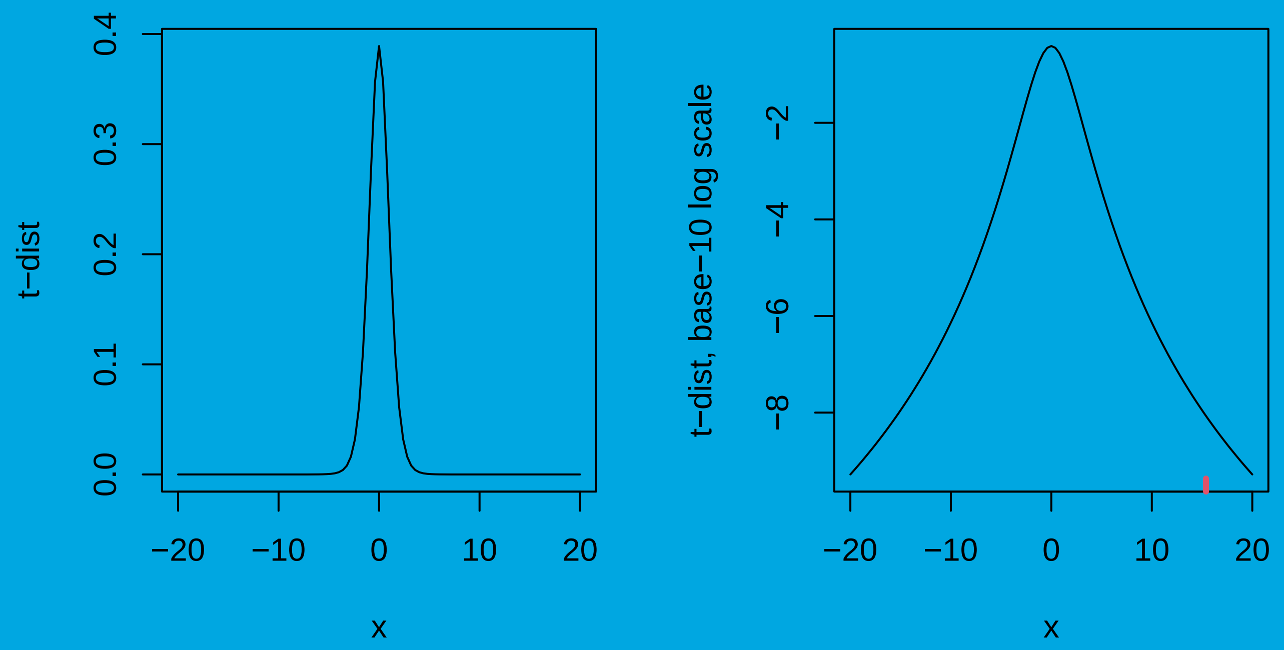
```
par(mfrow=c(1,2)); hist(ratios)
qqplot(ratios, qt((1:9999)/10000, df = n-2),
    ylab="quantiles", xlab="ordered t-ratios")
abline(0,1)
```



... additional evidence that the t-ratios follow a t-distribution on 10 degrees of freedom ...

How reasonable is it that the true slope is 0?

```
obstRatio <- 2.015/.1309# observed t-ratio
par(mfrow=c(1,2))
curve(dt(x, df=10), -20, 20, ylab="t-dist")
curve(log(dt(x, df=10), base=10), -20, 20,
    ylab="t-dist, base-10 log scale")
rug(obstRatio, col=2, lwd=3) # locate the observed t-ratio
```

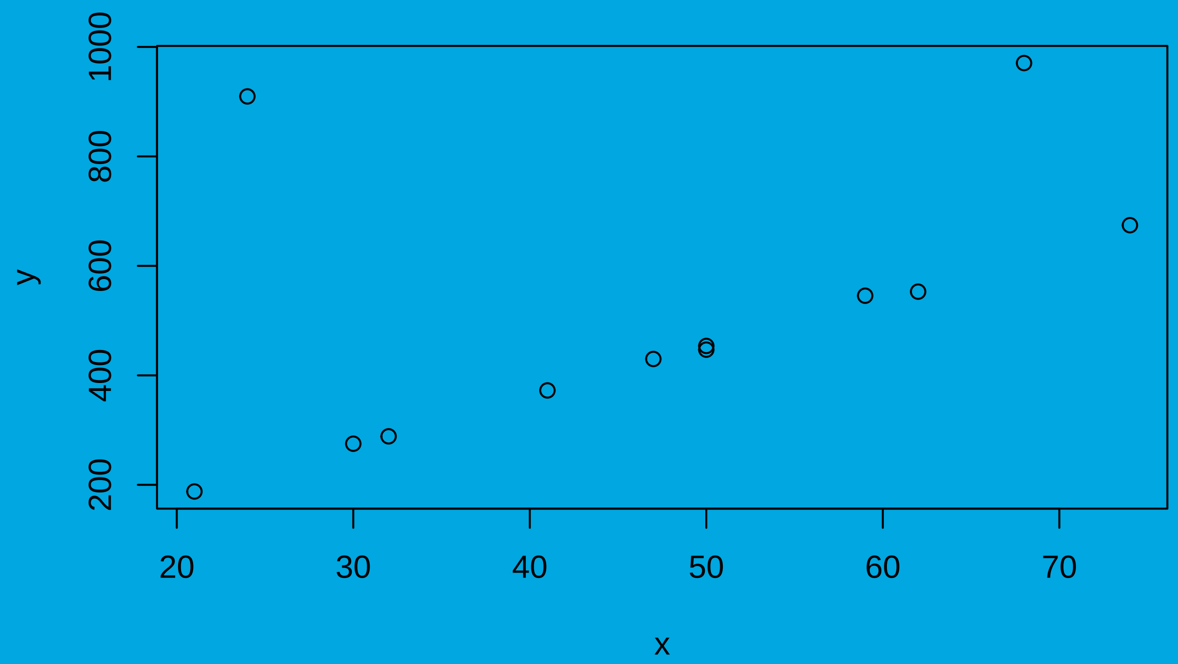


The probabilities are again so tiny that we use a base-10 log to visualize the probability that we could see such a large slope if the true slope were 0.

The probability, which is the area under the curve to the right of 15.349, is less than 10^{-7} – this is the p-value for the slope given on slide 26.

Simulated Linear Regression Data - Heavy Tailed Noise

```
x <- p2.12$temp
eps <- 3*rt(n, df=1.05) # scaled t on 1.05 degrees of freedom
y <- -6.332 + 9.208*x + eps
xy.df <- data.frame(x, y)
plot(y ~ x, data = xy.df)
```

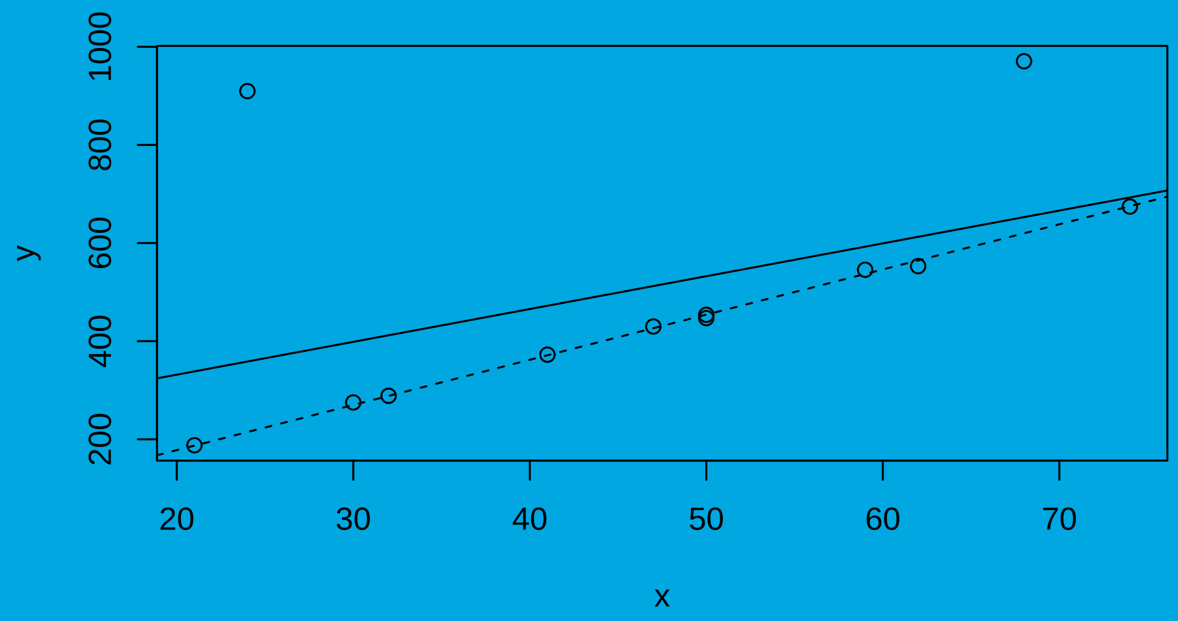


Simulated Linear Regression Data - Heavy Tailed Noise

```r
y.lm <- lm(y ~ x, data = xy.df)
# estimated beta0 hat and beta1 hat
coef(y.lm)

## (Intercept)           x
##  197.875753    6.689907

# estimated noise standard deviation
summary(y.lm)$sigma

## [1] 222.311
```

Plotting the Best Fit Line - Simulated

```
plot(y ~ x, data = xy.df)
abline(y.lm)
abline(-6.332, 9.208, lty=2)
```

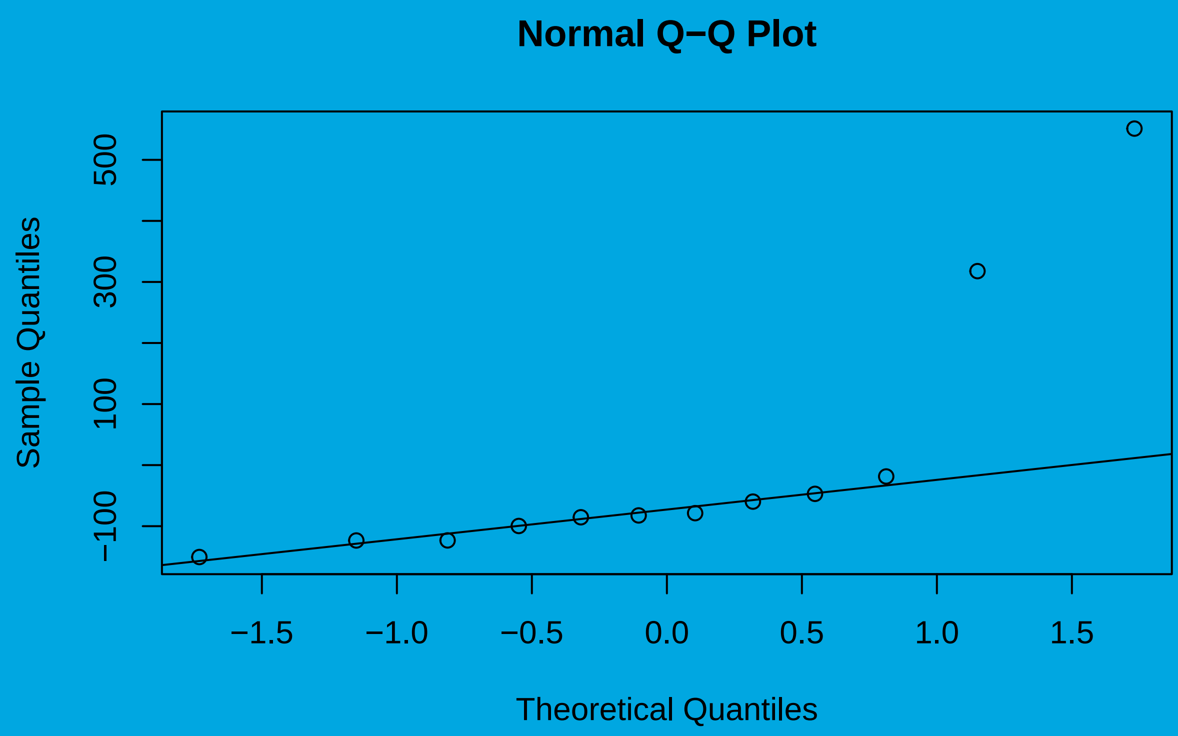


Dashed line: true line; Solid line: estimated line

Notice the effect of the *influential outlier* on the left. The slope of the fitted line is much smaller than the true slope. The outlier on the right is less influential since its magnitude is smaller. Both outliers have increased the intercept - a lot.

Simulated Linear Regression Data - Heavy Tailed Noise

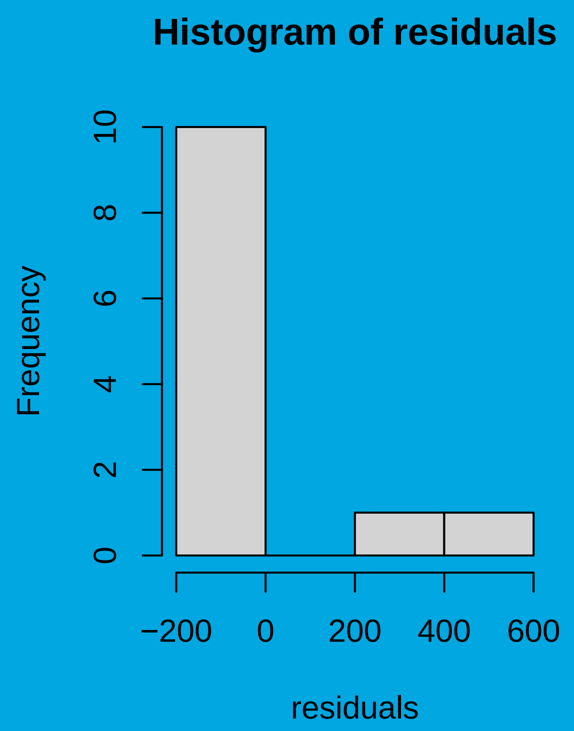
```r
residuals <- resid(y.lm)
qqnorm(residuals)
qqline(residuals)
```

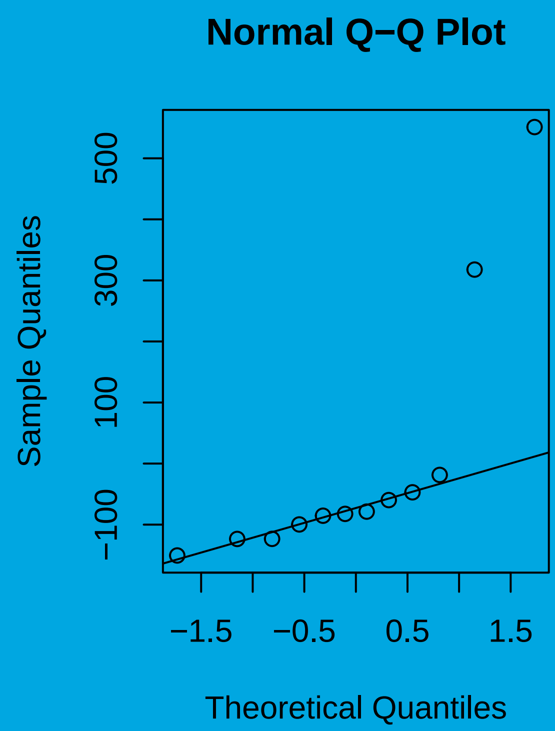


Clearly, not normal (which is correct, since the simulation is based on t errors, not normal errors)

How do the Heavy-Tailed Simulated Residuals Behave?

```r
residuals <- resid(y.lm)
par(mfrow=c(1,2))
hist(residuals)
qqnorm(residuals); qqline(residuals)
```



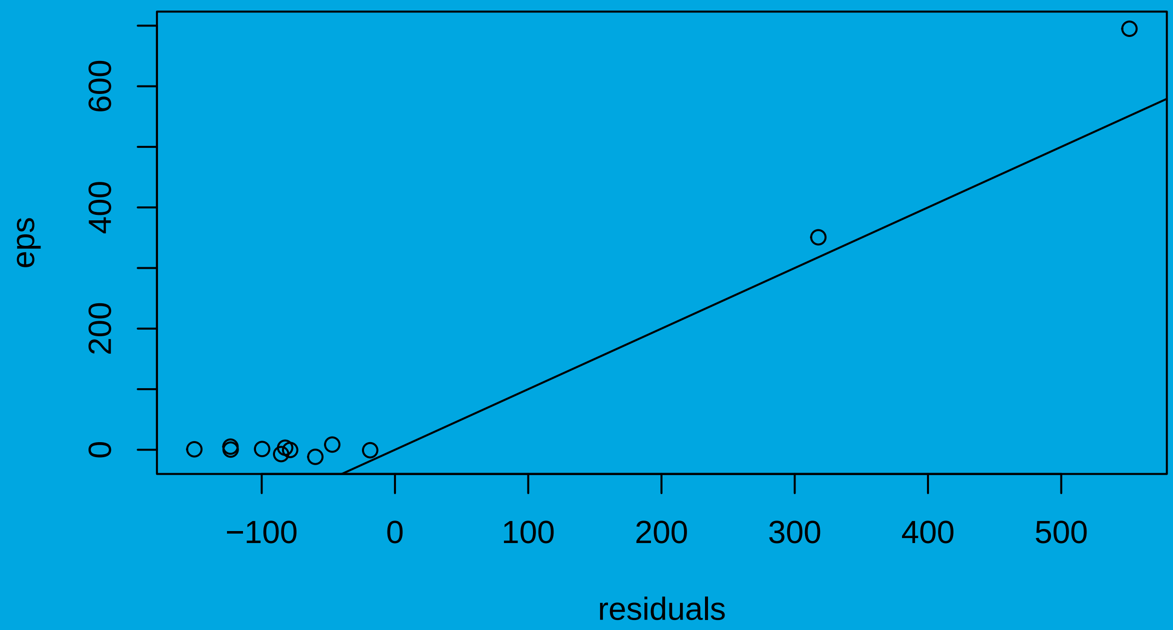


The residuals for the outliers are much higher than expected for normal data (not surprising, since this is not normal data).

How do the Heavy-Tailed Simulated Residuals Behave?

Compare the simulated residuals with the true errors:

```
plot(eps ~ residuals)
abline(0,1)
```

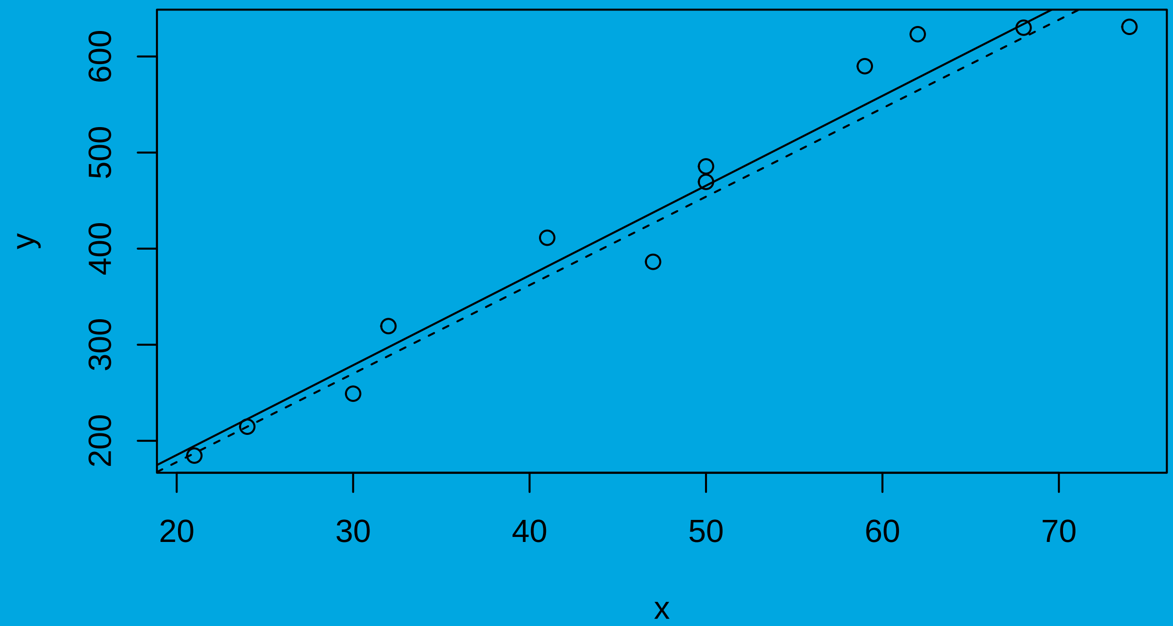


Simulated Linear Regression Data - Nonconstant Variance

```
# increasing variance
eps <- rnorm(n, sd=(x-15))
y <- -6.332 + 9.208*x +eps
xy.df <- data.frame(x,y)
y.lm <- lm(y ~ x, data = xy.df)
#estimated beta0 hat and beta1 hat
coef(y.lm)

## (Intercept)           x
##   -1.679628    9.345003

# estimated noise standard deviation (not valid!)
summary(y.lm)$sigma

## [1] 35.55287
```

Plotting the Best Fit Line - Simulated

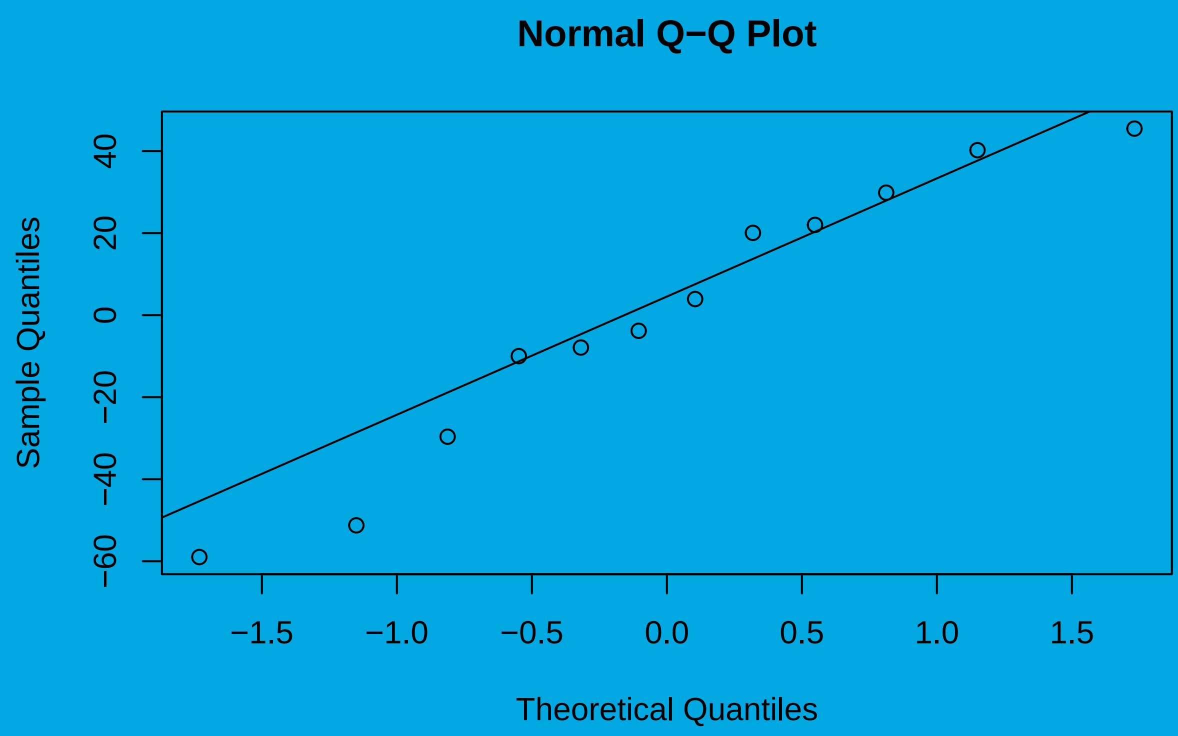
```r
plot(y ~ x, data = xy.df)
abline(y.lm)
abline(-6.332, 9.208, lty=2)
```



Dashed line: true line; Solid line: estimated line

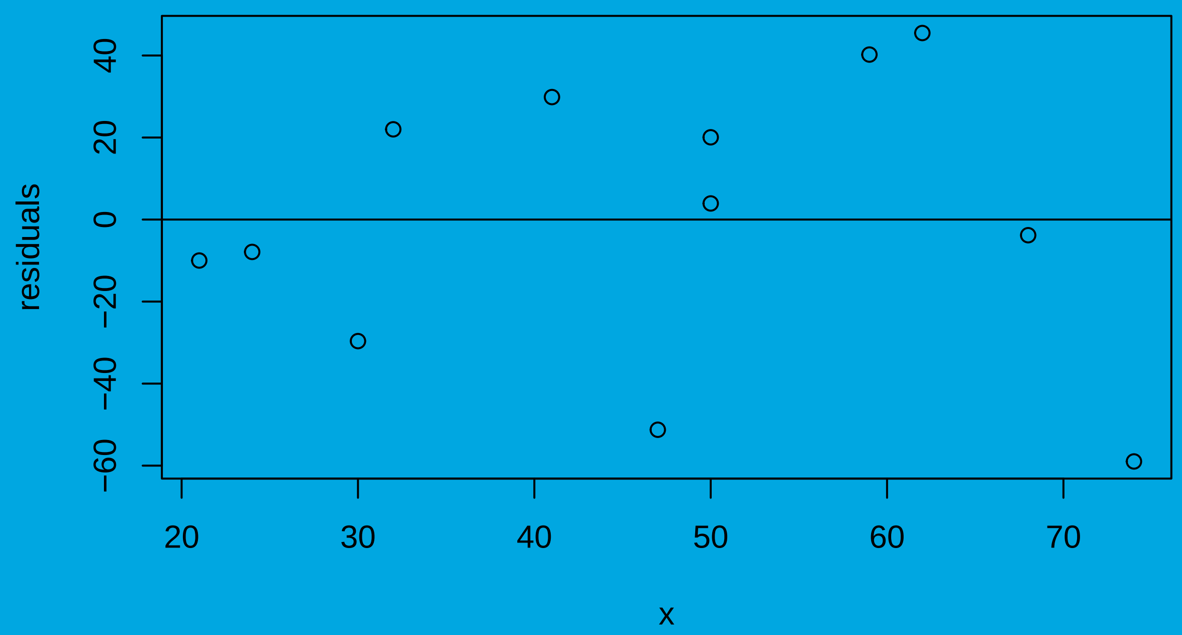
Simulated Linear Regression Data - Nonconstant Variance

```
residuals <- resid(y.lm)
qqnorm(residuals)
qqline(residuals)
```



Residuals: How do They Change with x?

```
plot(residuals ~ x , data = xy.df)
abline(h = 0)
```



When you see this kind of pattern, you should consider weighted least-squares. It will give improved estimates of the slope.

Ordinary (unweighted) Least-Squares Simulation:

```
Nsims <- 20000; slopes <- sderrors <- numeric(Nsims)
for (i in 1:Nsims) {# 20000 simulated data sets
   eps <- rnorm(n, sd=(x-15))
    y <- -6.332 + 9.208*x +eps
    xy.df <- data.frame(x,y)
    y.lm <- lm(y ~ x, data = xy.df)
    slopes[i] <- coef(y.lm)[2]
    sderrors[i] <- summary(y.lm)$coefficients[2,2]
}
mean(slopes); sd(slopes)

## [1] 9.206187
## [1] 0.6807442
```

Comparing Weighted Least-Squares with Ordinary Least-Squares

Weighted Least-Squares Simulation:

```
for (i in 1:Nsims) {# 20000 simulated data sets
   eps <- rnorm(n, sd=(x-15))
    y <- -6.332 + 9.208*x +eps
    xy.df <- data.frame(x,y)
    y.lm <- lm(y ~ x, data = xy.df, weights=1/(x-15))
    slopes[i] <- coef(y.lm)[2]
    sderrors[i] <- summary(y.lm)$coefficients[2,2]
}
mean(slopes); sd(slopes)

## [1] 9.213301
## [1] 0.5004347
```

The standard error of the slope estimate is less when weighted least-squares is used. So if there is evidence of a changing variance, you should try to use weights – if you can!