# Solutions to Exercise Series 2

**1.** **a)**
```
data(longley)
plot(Unemployed ~ GNP, data = longley, main = "longley data",
     sub = "USA")
```

**b)**
```
pop <- factor( longley$Population > 115 )
plot(Unemployed ~ GNP, data = longley, main = "longley data",
     sub = "USA", col = pop)
```

Alternatively, we can define the variable `pop` by

```
pop <- factor(ifelse(longley$Population < 115,
                     "Population < 115", "Population >= 115"))
```

or

```
pop <- ifelse( longley$Population < 115, 1, 2 )
```

**c)**
```
plot(Unemployed ~ GNP, data = longley, main = "longley data",
     sub = "USA", col = pop)
text(longley$GNP, longley$Unemployed, labels = longley$Year, pos = 1,
     cex = 0.7)
```

Note that we can reproduce the same plot using the package `ggplot2`. You need to install the package `ggplot2` before using it for the first time. This can be done with the command `install.packages("ggplot2")`.

```
require(ggplot2)
longley$pop <- factor( longley$Population > 115 )
g <- ggplot(data = longley, aes(x = GNP, y = Unemployed, color = pop)) +
  geom_point()
g + ggtitle("Longley Data Set") + geom_text(hjust = 0, vjust = 0,
                                            aes(label = Year))
```

**2.** See exercise sheet.

**3.** **Wilcoxon-test:**
There are two possibilities: a Wilcoxon signed-rank test (paired) or a Wilcoxon rank-sum test (one-sample).

```
## 1) Wilcoxon signed-rank test:
wilcox.test(bitumen[, "man"], bitumen[, "aut"],
            alternative = 'two.sided', paired = TRUE)
```

```
## 2) Wilcoxon rank-sum test:
wilcox.test(bitumen[, "diff"], mu = 0)
```

The test rejects the null hypothesis that both procedures are equal on the 5%-significance level.

**T-test:**
There are again two possibilities: either a paired t-test comparing the two methods or a one-sample t-test of the differences.

```
## 1) Paired t-test:
t.test(bitumen[, "man"], bitumen[, "aut"], alternative = 'two.sided',
       paired = TRUE)
```

```
## 2) One-Sample t-test:
t.test(bitumen[, "diff"], mu = 0, alternative = 'two.sided')
```

**Checking assumption of normality:**
Because a t-test assumes normality, we must check these assumptions before trusting the results of the tests.

```
par(mfrow = c(1,1))
#check if the differences are normally-distributed
qqnorm(bitumen[, "diff"]); qqline(bitumen[, "diff"])
```

**Conclusion:** Both tests claim differences between the manual procedure and the machine. Here, it is better to use a Wilcoxon-test since normality cannot be checked properly with such few observations.

**4.** **a)**
```
table(d.umwelt[, "Beeintr"], d.umwelt[, "Schule"])
```

**b)** The Chi-squared test clearly rejects the null hypothesis of independence of the two parameters with a p-value below $2.2 \cdot 10^{-16}$.

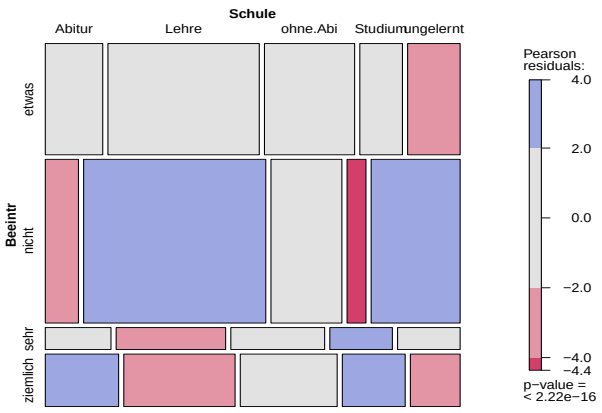
```
chisq.test(d.umwelt[, "Beeintr"], d.umwelt[, "Schule"])
```

**c)** The residuals show that especially people with low educational level feel much less affected than expected, whereas people with higher educational level feel more affected than expected under independence.

```
fit <- chisq.test(d.umwelt[, "Beeintr"], d.umwelt[, "Schule"])
fit$residuals
```

We visualize these results by using the `mosaic()`-plot function from the package `vcd` and set `shade = TRUE`:

```
require(vcd)
mosaic(~ Beeintr + Schule, shade = TRUE, data = d.umwelt)
```



**5.** **a)** Create the new variable `logpres`:

```
d.forbes$logpres <- log10(d.forbes$pres)
```

**b)**
```
plot(d.forbes$bp, d.forbes$logpres, xlab = "bp: Boiling Point",
     ylab = "logpres: log10(Pressure)")
```

There seems to be a linear relationship between the variables `bp` and `logpres`. The variables are highly correlated:

```
cor(d.forbes$logpres, d.forbes$bp)
## [1] 0.9974771
```

**c)** We add a regression line to the scatter plot of `logpres` against `bp`:

```
fit <- lm(logpres ~ bp, data = d.forbes)
plot(d.forbes$bp, d.forbes$logpres, xlab = "bp: Boiling Point",
     ylab = "logpres: log10(Pressure)")
abline(fit, col = 2)
```

An alternative to the command `abline(fit)`, which is much less convenient to type, is the following:

```
abline(a = coef(fit)["(Intercept)"], b = coef(fit)["bp"])
```

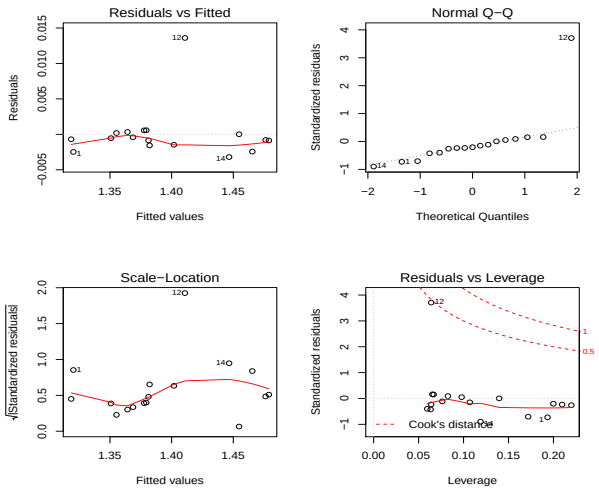
One observation lies much further away from the regression line than the others. You can identify the observation as follows:

- Type the following command into the console:

```
identify(d.forbes$bp, d.forbes$logpres)
```

- Click with the left mouse button on the suspect observation.
- Press ESC to exit the locator.

**d)**
```
par(mfrow = c(2,2))
plot(fit)
```




We can see that the observation 12 is an outlier. We cannot say if the assumptions are fulfilled or not, because of the extreme value.

**e)**
```
summary(fit)
```

The values of the test statistic ($T = 54.4$) for the (two-sided) test with the null hypothesis of zero slope can be gleaned from the summary-output. The corresponding p-value

($< 2 \cdot 10^{-16}$) is smaller than 0.05. Therefore we can reject the null hypothesis and consider $\beta_1$ as significantly different from 0.

**f)**
```
fit2 <- lm(logpres ~ bp, data = d.forbes[-12,])
summary(fit2)
```

Without the $12^{th}$ observation there is only a slight change of the estimated coefficients $\widehat{\beta_0}$ and $\widehat{\beta_1}$, but the corresponding standard errors $\text{se}^{(\hat{\beta_0})}$ and $\text{se}^{(\hat{\beta_1})}$ are much smaller. That means that $\beta_0$ and $\beta_1$ can be more accurately estimated. The variance of the residuals ($\widehat{\sigma}^2 = 0.0011^2 = 1.2 \times 10^{-6}$) is also much smaller than before ($\widehat{\sigma}^2 = 0.0038^2 = 1.4 \times 10^{-5}$).

## 6. Anscombe data

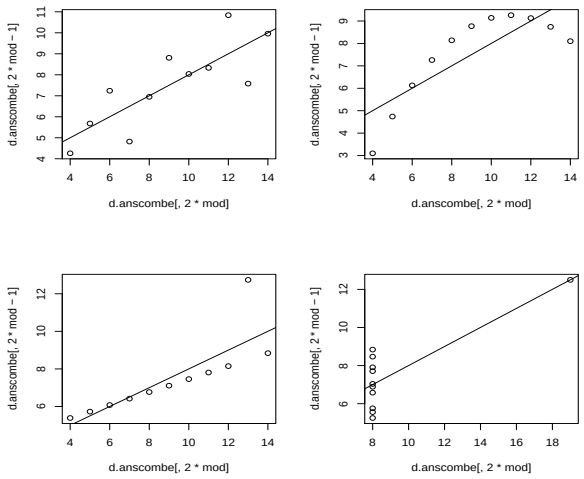
**a)** In order to compare the results, we create following table. We are using a `for` loop for illustration, yet there are many other ways to create it:

```
## first create the table and add row and column names
rnames <- c("Intercept", "Slope", "se(Intercept)", "se(Slope)",
            "Sigma^2", "R^2")
cnames <- paste("Mod", 1:4, sep = "")
t.table <- matrix(rep(NA,6*4), ncol = 4, dimnames = list(rnames, cnames))

## for each model, fit the regression and extract the values of interest
for (mod in 1:4) {
m <- lm(d.anscombe[,2*mod-1] ~ d.anscombe[,2*mod]) # regression
m.s <- summary(m) # summary object

t.table[,mod] <- c(m.s$coef[,c("Estimate", "Std. Error")],
                           m.s$sigma^2, m.s$r.squared)
}

t.table <- round(t.table, 3) # round values to 3 decimal places
t.table
```

The coefficients and standard errors are almost identical for all four models.

**b)** Scatter plots:

```
par(mfrow = c(2,2))  # split plot page into 4
for(mod in 1:4){
  plot(d.anscombe[,2*mod], d.anscombe[,2*mod-1])
  abline(a = t.table["Intercept",mod], b = t.table["Slope",mod])
}
```




**c)** Considering the four scatter plots, you can see that only in the first case it makes sense to fit a linear regression line. In the second case the relationship between the variables is not linear but appears to be more quadratic. In the third case there is an outlier which has a strong influence on the estimation of the parameters. In the fourth case the regression line depends only on one point.

**Conclusion:** It is not enough to only look at the `summary()` output. In all models this is approximately the same. It is crucial to perform a residual analysis for every fit you undertake.