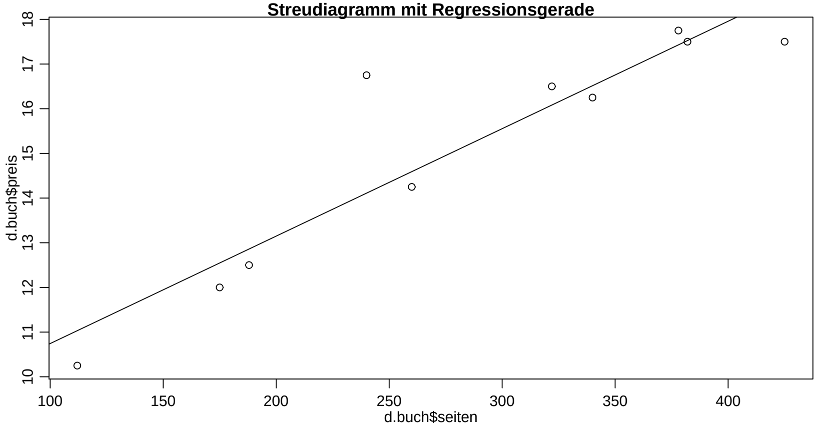
Weiterbildungs-Lehrgang in Angewandter Statistik ETHZ 2009/2011 21.09.09

Angewandte Regression — Musterlösungen zur Serie 1

1. x

a) Einlesen, Steudiagramm

```
> d.buch <- read.table("http://stat.ethz.ch/Teaching/Datasets/WBL//buchpreis.dat",h
> str(d.buch)
'data.frame': 10 obs. of  2 variables:
 $ preis : num  10.2 14.2 17.5 12 16.2 ...
 $ seiten: int  112 260 382 175 340 322 188 240 425 378
> plot(d.buch$seiten,d.buch$preis,main="Streudiagramm mit Regressionsgerade")
```



b) Regression, Summary

```
> r.lm <- lm(preis~seiten,data=d.buch)
> summary(r.lm)
lm(formula = preis ~ seiten, data = d.buch)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0593 -0.5000 -0.3031  0.2345  2.6399

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 8.338182   1.055604   7.899 4.78e-05 ***
seiten      0.024050   0.003534   6.806 0.000137 ***
---
```



```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.095 on 8 degrees of freedom
Multiple R-squared: 0.8527,Adjusted R-squared: 0.8343
F-statistic: 46.32 on 1 and 8 DF,  p-value: 0.0001370
```

Die Gleichung der Regressionsgeraden lautet: `preis=8.34+0.024*seiten`.

c) Regressionsgerade siehe Teilaufgabe a)

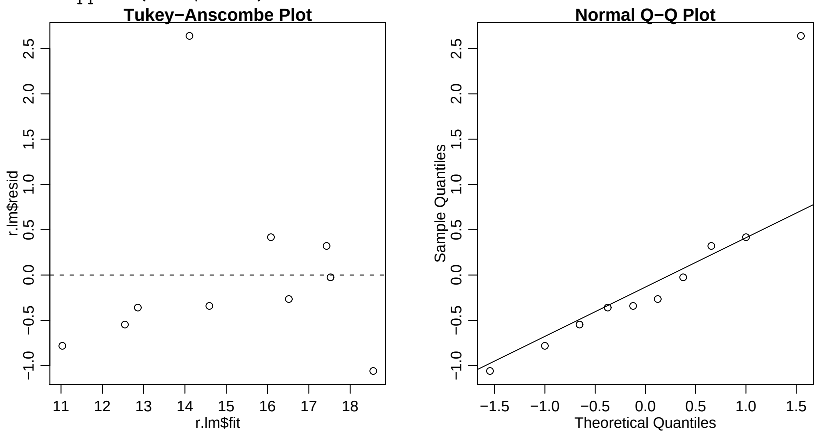
```
> abline(r.lm)
```

Test der Nullhypothese: "Der Verkaufspreis wird von der Seitenzahl nicht beeinflusst". Aufgrund des sehr kleinen p-Werts von 0.00014 wird die Nullhypothese auf dem 5%-Niveau deutlich verworfen. Die Seitenanzahl hat einen signifikanten Einfluss auf den Verkaufswert, was die sehr kleine Schätzung der Steigung nicht unbedingt erwarten lässt.

d) Preispolitik des Verlegers: Ein Buch hat einen Grundpreis von etwa 8.34 Dollar und Bücher mit vielen Seiten sind teuerer. Der Preisanstieg mit den Seiten ist aber klein (2.40 Dollar pro 100 Seiten).
Wohl würden nur wenige Leser dicke und noch dazu viel teuerere Bücher kaufen ...

e) Residuenanalyse, Aussreisser

```
#Residuenanalyse
> par(mfrow=c(2,2))
> plot(r.lm)
##oder
> par(mfrow=c(2,2))
> plot(r.lm$fit,r.lm$resid,main="Tukey-Anscomb Plot")
> abline(h=0,lty=2)
> qqnorm(r.lm$resid)
> qqline(r.lm$resid)
```




```
##Ausreiser-Identifikation
> par(mfrow=c(1,1))
> identify(qqnorm(r.lm$resid))
```

Beobachtung 8 fällt aus dem Rahmen und müsste überprüft werden.

f) Geben Sie ein 95%-Vertrauensintervall für die Steigung an:

```
> coef <- summary(r.lm)$coef
> coef[2,1]+c(-1,1)*coef[2,2]*qt(0.975,df=nrow(d.buch)-2)
[1] 0.01590124 0.03219810
```

g) Preis-Prognose für ein 600-seitiges Buch

```
> coef[1,1]+600*coef[2,1]
[1] 22.76799
##oder
> predict(r.lm,newdata=data.frame(seiten=600))
[1] 22.76799
> (t.range <- range(d.buch$seiten))
[1] 112 425
```

Ein 600-seitiges Buch würde etwas 22.80 Dollar kosten. Allerdings ist nicht klar, dass sich der Buchpreis für so gropsse Seitenzahlen auch linear verhält. Da das Modell nur auf Beobachtungen zwischen 112 und 425 Seiten gründet, ist eine solche Extrapolation zumindest heikel.

h)
- In Worten:
  Vertrauensband: (siehe Skript 2.3.g, 2.4.c und 2.4.d) Das Vertrauensband gibt an, wo die *idealen Funktionswerte* $h\langle\rangle$, also die Erwartungswerte von Y bei gegebenem x liegen.
  Vorhersageband: (siehe 2.4.d) Das Vorhersageband (Prognoseband) gibt an, wo eine zukünftige Beobachtung Y bei gegebenem x liegen.
- Rechnung von Hand:
  Vertrauensintervall: (siehe Skript 2.4.b)

$$
\begin{aligned}
(\hat{\alpha}+\hat{\beta}x_0) \pm q_{0.975}^{t_{n-2}} \mathrm{se}^{(\eta)} &= (\hat{\alpha}+\hat{\beta}x_0) \pm q_{0.975}^{t_{n-2}} \cdot \hat{\sigma} \sqrt{\frac{1}{n} + \frac{(x_0-\overline{x})^2}{SSQ^{(X)}}} \\
&= \hat{\alpha}+\hat{\beta}x_0 \pm q_{0.975}^{t_{n-2}} \cdot \sqrt{\frac{\hat{\sigma}^2}{n} + (x_0-\overline{x})^2 \frac{\hat{\sigma}^2}{SSQ^{(X)}}} \\
&= \hat{\alpha}+\hat{\beta}x_0 \pm q_{0.975}^{t_{n-2}} \cdot \sqrt{\frac{(\hat{\sigma})^2}{n} + (x_0-\overline{x})^2 (se^{(\beta)})^2}.
\end{aligned}
$$

  Setzen wir die Zahlen $n = 10$, $q_{0.975}^{t_8} = 2.306$ (benütze dazu R), $\hat{\sigma} = 1.095$, $\overline{x} = 282.2$ und $se^{(\beta)} = 0.004$, so erhalten wir

$$16.498 \pm 0.927$$

  was das intervall in "gerunden" $[15.588, 17.442]$ ergibt. Für $\mathrm{se}^{(\eta)}$ erhalten wir 0.4020669.
  Vorhersageintervall: (siehe 2.4.e*)

$$(\hat{\alpha}+\hat{\beta}x_0) \pm q_{0.975}^{t_{n-2}} \sqrt{\hat{\sigma}^2 + (\mathrm{se}^{(\eta)})^2}$$

  Setzen wir wiederum die Zahlen ein, so erhalten wir das intervall ("gerunden") $[13.825, 19.205]$.
- Rechnung mit R
  Vertrauensintervall: R-Befehl:


```
> t.predict <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=340))
> t.ywert <- t.predict$fit
> t.sd <- t.predict$se.fit
> t.degree <-t.predict$df
> t.resscale <- t.predict$residual.scale
> t.vert.yu <- t.ywert-qt(0.975,df=t.degree)*t.sd
> t.vert.yo <- t.ywert+qt(0.975,df=t.degree)*t.sd
```

  was 15.58790 und 17.44224 ergibt.
  Vorhersageintervall:

```
> t.sqrt <- sqrt(t.sd**2+t.resscale**2)
> t.vorh.vo <- t.ywert-qt(0.975,df=t.degree)*t.sqrt
> t.vorh.v1 <- t.ywert+qt(0.975,df=t.degree)*t.sqrt
```

  was 13.825 und 19.205 ergibt.
  Bemerkung: Die Funktion `predict` beinhaltet schon die obigen Berechungen. Mit `?predict.lm` erhält man die nöptigen Informationen:

```
> ?predict
predict(object, newdata, se.fit = FALSE, scale = NULL, df = Inf,
             interval = c("none", "confidence", "prediction"),
             level = 0.95, type = c("response", "terms"),
             terms = NULL, na.action = na.pass,
             pred.var = res.var/weights, weights = 1, ...)
```

  Somit programmieren wir

```
> t.vert <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=340),
           interval ="confidence")
> t.vorh <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=340),
           interval = "prediction")
```

  mit dem R-Output

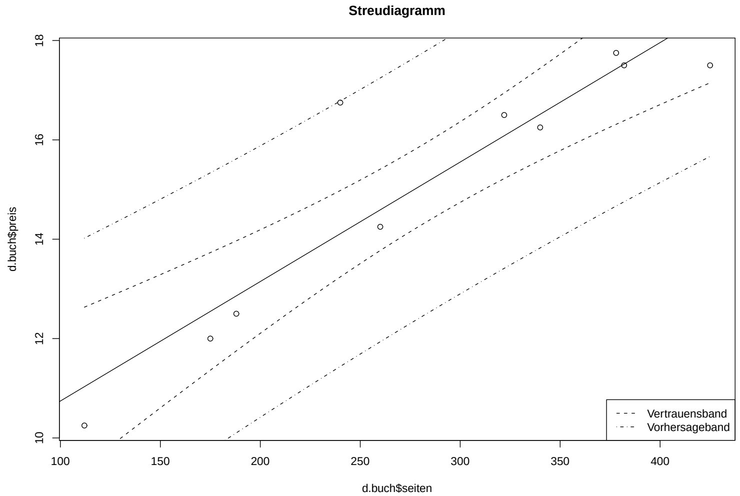
```
>      fit      lwr      upr
1 16.51507 15.58790 17.44224
>      fit      lwr      upr
1 16.51507 13.82475 19.20539
```

- Graphik
  Vertrauensband und Vorhersageband:

```
t.xwerte <- seq(t.range[1],t.range[2],by=1)
t.predict <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=t.xwerte))
t.ywert <- t.predict$fit
t.sd <- t.predict$se.fit
t.degree <-t.predict$df
t.resscale <- t.predict$residual.scale
t.vert.yu <- t.ywert-qt(0.975,df=t.degree)*t.sd
t.vert.yo <- t.ywert+qt(0.975,df=t.degree)*t.sd

t.sqrt <- sqrt(t.sd**2+t.resscale**2)
t.vorh.vo <- t.ywert-qt(0.975,df=t.degree)*t.sqrt
t.vorh.v1 <- t.ywert+qt(0.975,df=t.degree)*t.sqrt
plot(d.buch$seiten,d.buch$preis,main="Streudiagramm")
abline(r.lm)
lines(x=t.xwerte,y=t.vert.yu,lty=2)
```

```
lines(x=t.xwerte,y=t.vert.yo,lty=2)
lines(x=t.xwerte,y=t.vorh.vo,lty=4)
lines(x=t.xwerte,y=t.vorh.v1,lty=4)

legend("bottomright", c("Vertrauensband", "Vorhersageband"),
       lty=c(2,4), cex=1)
```

Dies ergibt die Graphik:



Bemerkung: Dieselben Funktionen resp Graphiken können eleganter mit `predict` geschrieben werden:

```
t.range <- range(d.buch$seiten)
t.xwerte <- seq(t.range[1],t.range[2],by=1)
t.vert <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=t.xwerte),
            interval ="confidence")$fit
t.vorh <- predict(r.lm,se.fit=T,newdata=data.frame(seiten=t.xwerte),
            interval ="prediction")$fit
plot(d.buch$seiten,d.buch$preis,main="Streudiagramm")
abline(r.lm)
lines(x=t.xwerte,y=t.vert[,2],lty=2)
lines(x=t.xwerte,y=t.vert[,3],lty=2)
lines(x=t.xwerte,y=t.vorh[,2],lty=4)
lines(x=t.xwerte,y=t.vorh[,3],lty=4)
legend("bottomright", c("Vertrauensband", "Vorhersageband"),
       lty=c(2,4), cex=1)
```

2. a) Bei allen vier Modellen sind der Achsenabschnitt, die Steigung und die zugehörigen Standardfehler, sowie $\widehat{\sigma}^2$ und R^2 praktisch identisch.

	Mod1	Mod2	Mod3	Mod4
Achsenab.	3.000	3.001	3.002	3.002
Steigung	0.500	0.500	0.500	0.500
se(Achsenab.)	1.125	1.125	1.124	1.124
se(Steigung)	0.118	0.118	0.118	0.118
Sigma^2	1.529	1.531	1.528	1.527
R^2	0.667	0.666	0.666	0.667

R-Code für die Erzeugung der Tabelle:

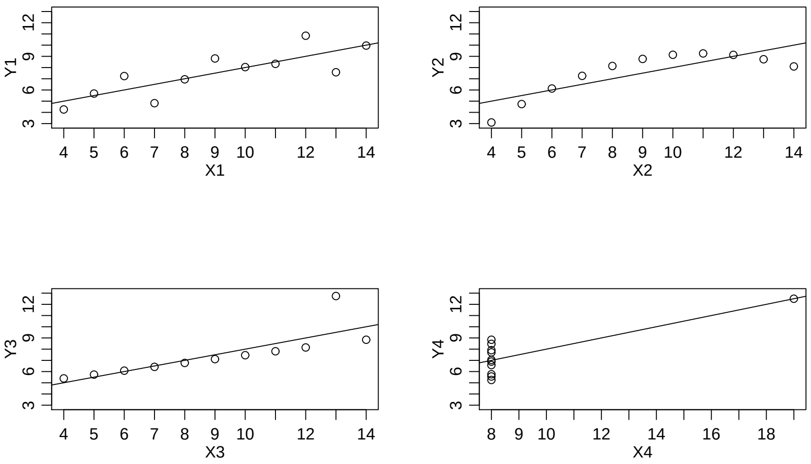
```
t.ans1 <- summary(lm(Y1 ~ X1, d.anscombe))
t.ans2 <- summary(lm(Y2 ~ X2, d.anscombe))
t.ans3 <- summary(lm(Y3 ~ X3, d.anscombe))
t.ans4 <- summary(lm(Y4 ~ X4, d.anscombe))

t.tabelle <- data.frame(Mod1 = c(t.ans1$coef[,c('Estimate','Std. Error')],
                                 t.ans1$sigma^2,t.ans1$r.squ),
                        Mod2 = c(t.ans2$coef[,c('Estimate','Std. Error')],
                                 t.ans2$sigma^2,t.ans2$r.squ),
                        Mod3 = c(t.ans3$coef[,c('Estimate','Std. Error')],
                                 t.ans3$sigma^2,t.ans3$r.squ),
                        Mod4 = c(t.ans4$coef[,c('Estimate','Std. Error')],
                                 t.ans4$sigma^2,t.ans4$r.squ))
t.tabelle <- round(t.tabelle,3)
rownames(t.tabelle) <- c('Achsenab.','Steigung','se(Achsenab.)',
                         'se(Steigung)','Sigma^2','R^2')
```

Um die Namen der Einträge im Objekt `t.ans1` abzufragen, benutzt man den Befehl `names()`:

```
> names(t.ans1)
 [1] "call"          "terms"         "residuals"     "coefficients"
 [5] "aliased"       "sigma"         "df"            "r.squared"
 [9] "adj.r.squared" "fstatistic"    "cov.unscaled"
```

b) R-Code für Sreudiagramme:

```
par(mfrow=c(2,2))
attach(d.anscombe)
plot(X1,Y1,ylim=c(3,13)) ; abline(t.ans1)
plot(X2,Y2,ylim=c(3,13)) ; abline(t.ans2)
plot(X3,Y3,ylim=c(3,13)) ; abline(t.ans3)
plot(X4,Y4,ylim=c(3,13)) ; abline(t.ans4)
detach(d.anscombe)
```



c) Betrachtet man die vier Streudiagramme, so sieht man, dass nur im ersten Fall eine lineare Regression angebracht ist. Im zweiten Fall ist die Beziehung zwischen X und Y nicht linear, sondern eher quadratisch. Im dritten Fall gibt es einen Ausreisser, welcher die geschätzten Parameter stark beeinflusst. Im vierten Fall wird die Regressionsgerade durch einen einzigen Punkt bestimmt.

Fazit: Es genügt **nicht**, nur $\widehat{\alpha}$, $\widehat{\beta}$, $se^{(\hat{\alpha})}$, $se^{(\hat{\beta})}$, R^2 und $\widehat{\sigma}$ anzuschauen. In allen Modellen sind diese Schätzungen fast gleich, aber die Datensätze sehen auf der Grafik ganz unterschiedlich aus.

3. **a)** R-Befehle:

```
t.X <- c(0,3,4,8,10,11,13,16,17,20)
t.E <- matrix(rnorm(10*100,sd=sqrt(2)),ncol=100)
t.Y <- 4+2*t.X+t.E
```

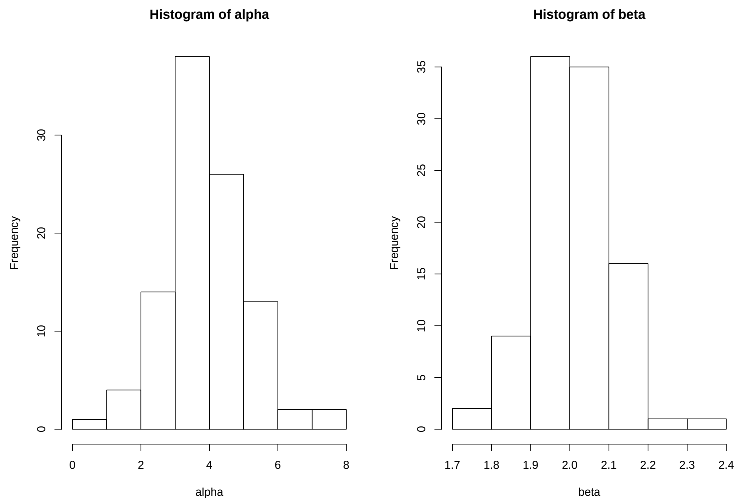
b) Re-Befehle:

```
r.coeff <- apply(t.Y,2,FUN=function(y) lm(y~t.X)$coefficients)
alpha <- r.coeff[1,]
beta <- r.coeff[2,]
```

c) R-Befehle:

```
par(mfrow=c(1,2))
hist(alpha)
hist(beta)
par(op)
```


Die Gipfel der Verteilungen liegen am richtigen Ort. Für α (=`ALPHA`) bei 4 und für β (=`BETA`) bei 2.



Bemerkung: Aus der Theorie wissen wir, dass

$$\widehat{\alpha} \sim \mathcal{N}\big(\alpha,\ \sigma^2(\alpha)\big), \qquad \sigma^2(\alpha) = \sigma^2 \left(\frac{1}{n} + \overline{x}^2/\mathrm{SS}_X\right)$$

und

$$\widehat{\beta} \sim \mathcal{N}(\beta,\ \sigma^2(\beta)), \qquad \sigma^2(\beta) = \sigma^2/\mathrm{SS}_X$$

wobei SS_X eine Abkürzung für die sogenannte Quadratsumme der x-Werte ist. Mit $\mathrm{SS}_X = \sum_i (x_i - \bar{x})^2 = 383.6$ und $\bar{x} = 10.2$ ist also

$$\sigma^2(\alpha) = 2\left(\frac{1}{10} + 10.2^2/383.6\right) = 0.742, \qquad \sigma^2(\beta) = 2/383.6 = 0.0052$$

d.h.

$$\widehat{\alpha} \sim \mathcal{N}(4,\ 0.742) \quad \text{und} \quad \widehat{\beta} \sim \mathcal{N}(2,\ 0.0052)$$

Die Simulation ist schon mit 100 Stichproben recht gut.