WDDA FS 2026: Leitfaden für Aufgabenserie 5

2026-02-13

1 Einleitung

Dieser Leitfaden bietet detaillierte Erklärungen für die Übungen in WDDA FS 2026 Aufgabenserie/Übungsblatt 5. Für jede Aufgabe werden wir den Denkprozess (und die notwendigen Schritte in R) durchgehen und erklären, wie man zur richtigen Lösung gelangt.

2 Aufgabe 1: Advertising–Datensatz (TV → Sales)

Aufgabenstellung: Betrachten Sie den Datensatz **Advertising** mit `TV` als erklärende Variable. Welche der folgenden Geraden passt am besten zu den Daten?

(a) Intercept = 7.1, Steigung = 0.049
(b) Intercept = 6.8, Steigung = 0.048
(c) Intercept = 7.0, Steigung = 0.045
(d) Intercept = 7.3, Steigung = 0.041

2.1 Schritt 1: Daten einlesen

```r
# Der Pfad zur Datei muss in Ihrem fall allenfalls angepasst werden
adv <- read_excel("../data/WDDA_05.xlsx", sheet = "Advertising")
head(adv)
```

```
## # A tibble: 6 x 4
##      TV radio newspaper sales
##   <dbl> <dbl>     <dbl> <dbl>
## 1 230.   37.8      69.2  22.1
## 2  44.5  39.3      45.1  10.4
## 3  17.2  45.9      69.3   9.3
## 4 152.   41.3      58.5  18.5
## 5 181.   10.8      58.4  12.9
## 6   8.7  48.9      75     7.2
```

2.2 Schritt 2: RSS für jede Gerade berechnen

Wir definieren jede Kandidaten-Gerade und berechnen die Residuenquadratsumme (RSS):

```r
tv    <- adv$TV
sales <- adv$sales

kandidaten <- list(
  a = c(intercept = 7.1, slope = 0.049),
  b = c(intercept = 6.8, slope = 0.048),
  c = c(intercept = 7.0, slope = 0.045),
  d = c(intercept = 7.3, slope = 0.041)
)
```

```r
rss <- sapply(kandidaten, function(par) {
  pred  <- par["intercept"] + par["slope"] * tv
  sum((sales - pred)^2)
})
print(rss)
```

```
##        a        b        c        d
## 2121.642 2108.255 2144.869 2261.464
```

2.3 Schritt 3: Beste Gerade auswählen

Die Gerade mit dem kleinsten RSS passt am besten. Aus der Ausgabe wählen wir **(b)**:

> **Beste Wahl:** Intercept = 6.8, Steigung = 0.048

3 Aufgabe 2: Diamond Rings (Price ~ Weight)

Aufgabenstellung: Analysieren Sie den Zusammenhang zwischen Gewicht (`weight`) und Listenpreis (`price`) von Diamantringen.

1. Streudiagramm;
2. Lineares Modell und Interpretation von Intercept & Steigung;
3. Interpretation von R^2, RSE, RSS & TSS;
4. Geschätzter Preisunterschied zwischen 0.25 und 0.35 ct;
5. Modell in CHF (1 SGD = 0.68 CHF) umrechnen;
6. Einfluss der Fixkosten;
7. Ring mit 0.18 ct für SGD 325: Schnäppchen?;
8. Residuen plotten und Standardabweichung interpretieren.

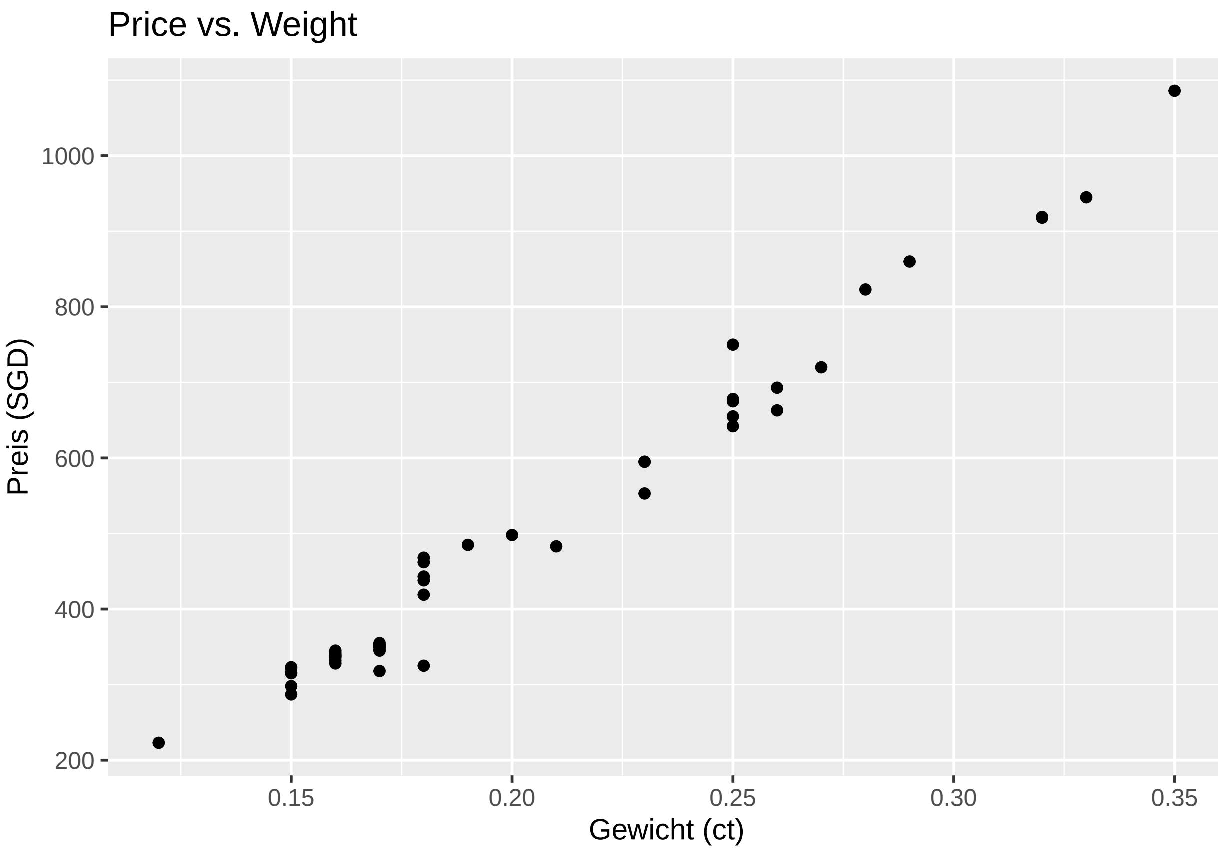
3.1 Schritt 1: Daten einlesen

```r
diamonds <- read_excel("../data/WDDA_05.xlsx", sheet = "Diamonds Rings") %>%
  rename(weight = `Weight (carats)`,
         price  = `Price (Singapore dollars)`)
head(diamonds)
```

```
## # A tibble: 6 x 2
##   weight price
##    <dbl> <dbl>
## 1   0.17   355
## 2   0.16   328
## 3   0.17   350
## 4   0.18   325
## 5   0.25   642
## 6   0.16   342
```

3.2 Schritt 2: Streudiagramm

```r
ggplot(diamonds, aes(x = weight, y = price)) +
  geom_point() +
  labs(title = "Price vs. Weight", x = "Gewicht (ct)", y = "Preis (SGD)")
```



3.3 Schritt 3: Lineares Modell schätzen

```
mod_dr <- lm(price ~ weight, data = diamonds)
summary(mod_dr)
```

```
##
## Call:
## lm(formula = price ~ weight, data = diamonds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.159 -21.448  -0.869  18.972  79.370
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -259.63      17.32  -14.99   <2e-16 ***
## weight       3721.02      81.79   45.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.84 on 46 degrees of freedom
## Multiple R-squared:  0.9783, Adjusted R-squared:  0.9778
## F-statistic:  2070 on 1 and 46 DF,  p-value: < 2.2e-16
```

- **Intercept** $\approx$ -259.63: theoretischer Preis bei 0 ct (nicht sinnvoll).
- **Steigung** $\approx$ 3721.02: Mehrpreis von SGD 3721.02 pro zusätzlichem Karat.

3.4 Schritt 4: R^2, RSE, RSS, TSS berechnen

```
resid_dr <- resid(mod_dr)
rss_dr   <- sum(resid_dr^2)
tss_dr   <- sum((diamonds$price - mean(diamonds$price))^2)
rse_dr   <- sqrt(rss_dr / df.residual(mod_dr))
r2_dr    <- summary(mod_dr)$r.squared

cat("R^2 =", round(r2_dr, 4), "\n")
```

```
## R^2 = 0.9783
```

```
cat("RSE =", round(rse_dr, 2), "SGD\n")
```

```
## RSE = 31.84 SGD
```

```
cat("RSS =", round(rss_dr, 0), "SGD^2\n")
```

```
## RSS = 46636 SGD^2
```

```
cat("TSS =", round(tss_dr, 0), "SGD^2\n")
```

```
## TSS = 2145232 SGD^2
```

3.5 Schritt 5: Preisunterschied für 0.25 → 0.35 ct

$$\Delta \hat{price} = b_1 \times (0.35 - 0.25)$$

```
preisdiff <- coef(mod_dr)["weight"] * (0.35 - 0.25)
cat("Geschätzter Unterschied:", round(preisdiff, 1), "SGD\n")
```

```
## Geschätzter Unterschied: 372.1 SGD
```

3.6 Schritt 6: Modell in CHF umrechnen

$$\hat{price}_{CHF} = 0.68 \times \hat{price}_{SGD}$$

```
b0_chf <- coef(mod_dr)["(Intercept)"] * 0.68
b1_chf <- coef(mod_dr)["weight"]       * 0.68
cat("Preismodell (CHF): ŷ =", round(b0_chf,1), "+", round(b1_chf,1), "× weight\n")
```

```
## Preismodell (CHF): ŷ = -176.5 + 2530.3 × weight
```

3.7 Schritt 7: Einfluss der Fixkosten

Fixkosten erhöhen den **Intercept**, da sie den Basispreis auch bei 0 ct erhöhen. Die Steigung bleibt unverändert.

3.8 Schritt 8: Schnäppchen-Check für 0.18 ct und SGD 325

```
pred_018 <- predict(mod_dr, newdata = data.frame(weight = 0.18))
ci      <- predict(mod_dr, newdata = data.frame(weight = 0.18),
                   interval = "prediction", level = 0.95)
cat("Prognose für 0.18 ct:", round(pred_018,1), "SGD\n")
```

```
## Prognose für 0.18 ct: 410.2 SGD
```

```r
cat("95% Prognose-Intervall: [", round(ci[,"lwr"],1), ",", round(ci[,"upr"],1), "] SGD\n")
```

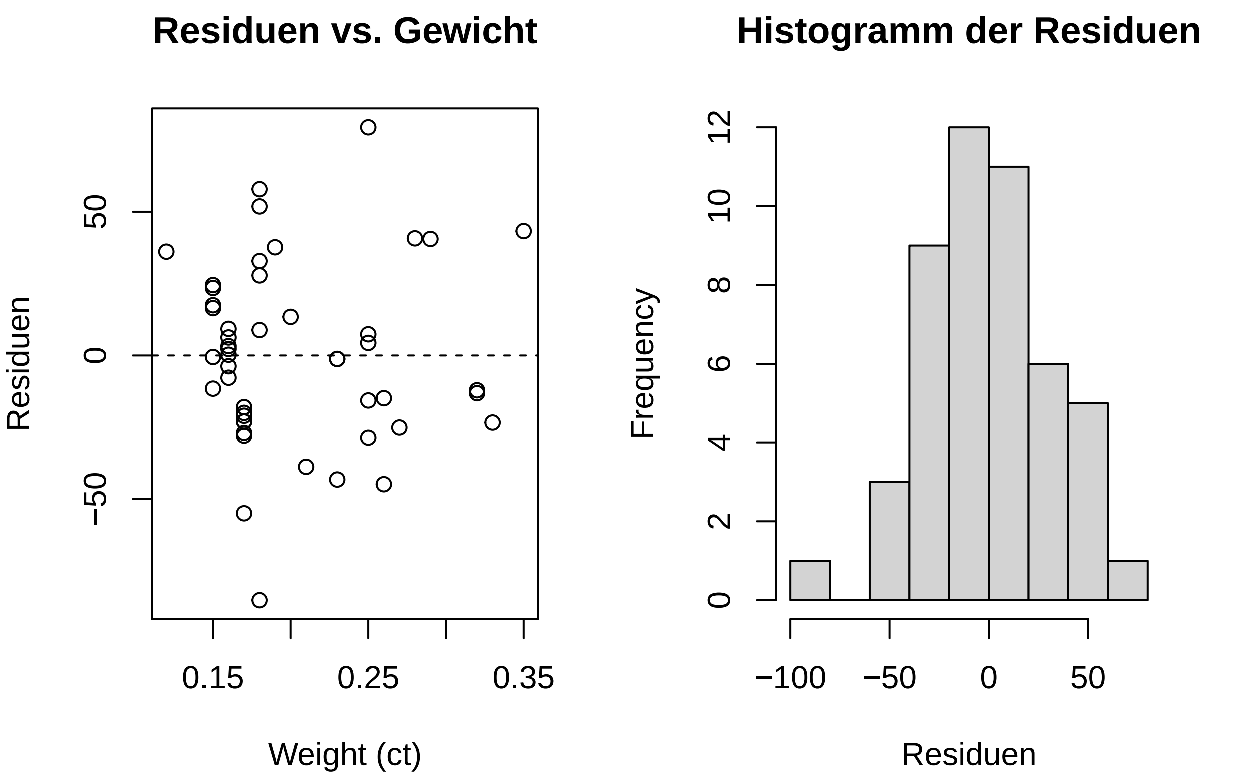
```
## 95% Prognose-Intervall: [ 345.3 , 475 ] SGD
```

Da SGD 325 unter dem Untergrenzwert liegt, ist es ein **Schnäppchen**.

3.9 Schritt 9: Residuen analysieren

```r
resid_dr <- resid(mod_dr)
mean_resid <- mean(resid_dr)
sd_resid   <- sd(resid_dr)

# Plot
par(mfrow = c(1,2))
plot(diamonds$weight, resid_dr,
     main = "Residuen vs. Gewicht", xlab = "Weight (ct)", ylab = "Residuen")
abline(h = 0, lty = 2)
hist(resid_dr, main = "Histogramm der Residuen",
     xlab = "Residuen")
```




```r
par(mfrow = c(1,1))
cat("Residuen-Mittelwert:", round(mean_resid,2), "\n")
```

```
## Residuen-Mittelwert: 0
```

```r
cat("Residuen-SD:", round(sd_resid,2), "SGD\n")
```

```
## Residuen-SD: 31.5 SGD
```

Die **Standardabweichung der Residuen** (31.84 SGD) zeigt den typischen Abstand der beobachteten Preise von der Regressionsgerade.

4 Aufgabe 3: Netzwerk-Performance (Download)

Aufgabenstellung: Untersuchen Sie den Datensatz **Download** mit Übertragungszeit (`time_sec`) und Dateigrösse (`size_mb`).

1. Streudiagramm;
2. Lineares Modell und Interpretation;
3. R^2, RSE, RSS & TSS;
4. Geschätzte Zeitdifferenz 50 → 60 MB;
5. Modell in Minuten und Kilobyte;
6. Residuen vs Grösse;
7. Residuen vs geschätzte Werte;
8. Datenmenge in 15 Sekunden abschätzen.

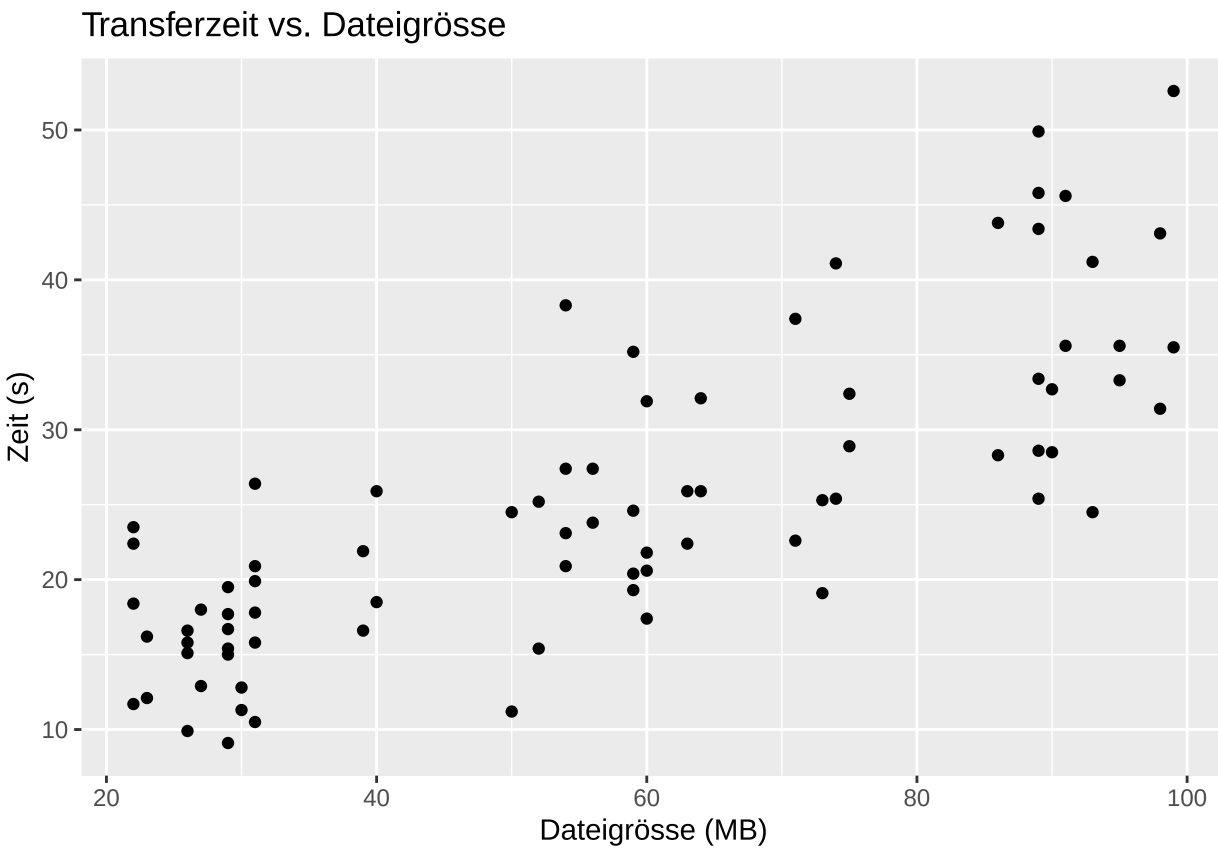
4.1 Schritt 1: Daten einlesen

```
dl <- read_excel("../data/WDDA_05.xlsx", sheet = "Download") %>%
  rename(time_sec = `Transfer Time (sec)`,
         size_mb  = `File Size (MB)`)
head(dl)
```

```
## # A tibble: 6 x 4
##   time_sec size_mb `Hours past 8` Vendor
##      <dbl>   <dbl>          <dbl> <chr>
## 1     18.4      22         0      MS
## 2     22.4      22         0.0625 NP
## 3     11.7      22         0.125  MS
## 4     23.5      22         0.188  NP
## 5     16.2      23         0.25   MS
## 6     12.1      23         0.312  NP
```

4.2 Schritt 2: Streudiagramm

```
ggplot(dl, aes(x = size_mb, y = time_sec)) +
  geom_point() +
  labs(title = "Transferzeit vs. Dateigrösse",
       x = "Dateigrösse (MB)", y = "Zeit (s)")
```



4.3 Schritt 3: Lineares Modell

```
mod_dl <- lm(time_sec ~ size_mb, data = dl)
summary(mod_dl)
```

```
##
## Call:
## lm(formula = time_sec ~ size_mb, data = dl)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.912  -4.671  -1.103   3.383  14.741
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.27466    1.71491   4.242 6.04e-05 ***
## size_mb      0.31331    0.02751  11.391  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.243 on 78 degrees of freedom
## Multiple R-squared:  0.6246, Adjusted R-squared:  0.6197
## F-statistic: 129.8 on 1 and 78 DF,  p-value: < 2.2e-16
```

- **Intercept** ≈ 7.27 s: Startlatenz im Netzwerk.
- **Steigung** ≈ 0.3133 s/MB: zusätzliche Zeit pro MB.

4.4 Schritt 4: Kennzahlen berechnen

```
resid_dl <- resid(mod_dl)
rss_dl   <- sum(resid_dl^2)
tss_dl   <- sum((dl$time_sec - mean(dl$time_sec))^2)
rse_dl   <- sqrt(rss_dl / df.residual(mod_dl))
r2_dl    <- summary(mod_dl)$r.squared

cat("R^2 =", round(r2_dl,4), "\n")
```

```
## R^2 = 0.6246
```

```
cat("RSE =", round(rse_dl,2), "s\n")
```

```
## RSE = 6.24 s
```

```
cat("RSS =", round(rss_dl,0), "s²\n")
```

```
## RSS = 3040 s²
```

```
cat("TSS =", round(tss_dl,0), "s²\n")
```

```
## TSS = 8098 s²
```

4.5 Schritt 5: Zeitdifferenz 50 → 60 MB

$$\Delta \widehat{time} = b_1 \times (60 - 50)$$

```
time_diff <- coef(mod_dl)["size_mb"] * 10
cat("Geschätzter Unterschied:", round(time_diff,2), "s\n")
```

```
## Geschätzter Unterschied: 3.13 s
```

4.6 Schritt 6: Modell in Minuten & Kilobyte

Wir setzen 1 MB = 1000 KB und Zeit in Minuten (/60):

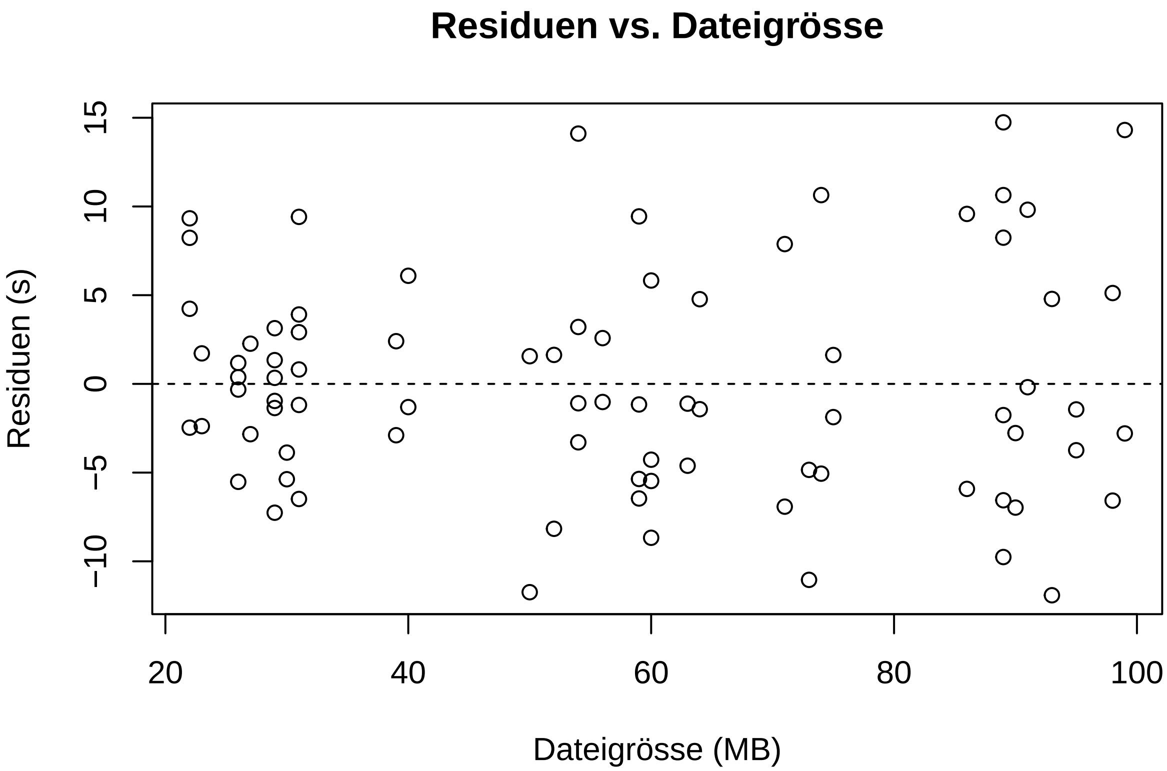
$$\begin{aligned}\widehat{time}_{min} &= \frac{7.2747}{60} + \frac{0.3133}{60} \times size_{MB} \\ &= 0.1212 + 0.005222 \times size_{MB}\end{aligned}$$

In Kilobyte:

$$\widehat{time}_{min} = 0.1212 + 5.22 \times 10^{-6} \times size_{KB}$$

4.7 Schritt 7: Residuen vs. Grösse

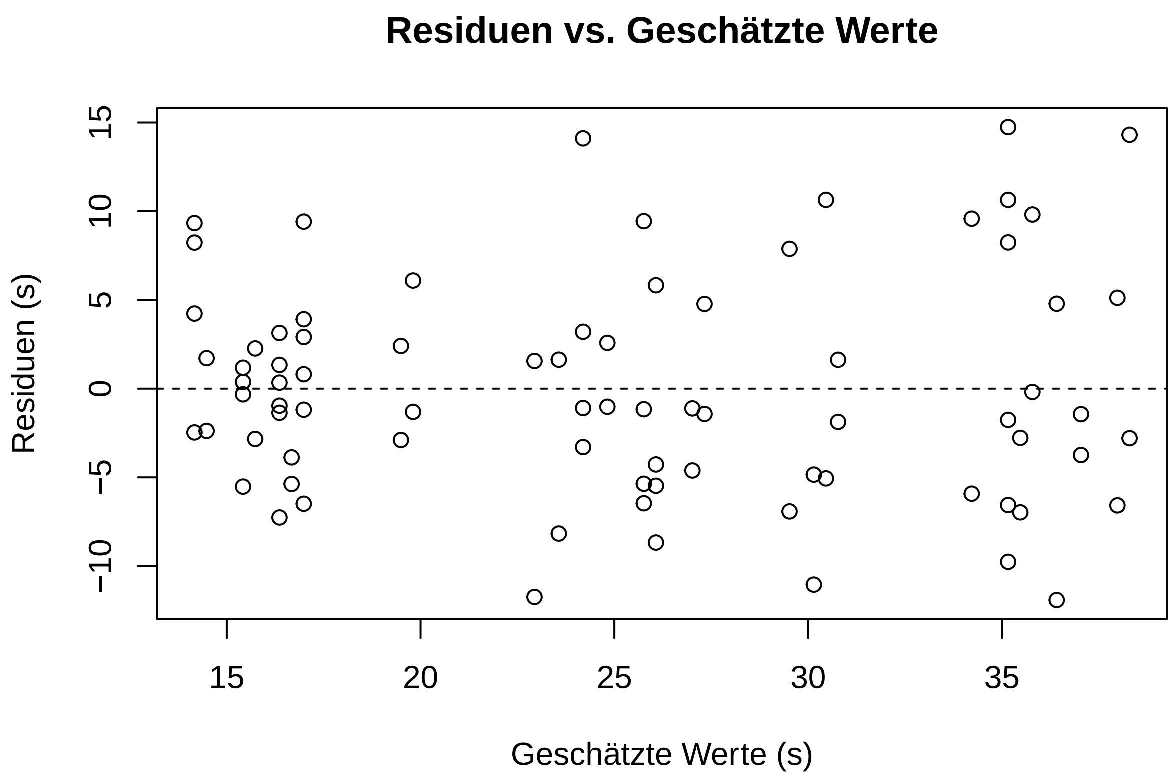
```
plot(dl$size_mb, resid_dl,
     main = "Residuen vs. Dateigrösse",
     xlab = "Dateigrösse (MB)", ylab = "Residuen (s)")
abline(h = 0, lty = 2)
```



Keine erkennbaren Muster – die Varianz scheint konstant.

4.8 Schritt 8: Residuen vs. geschätzte Werte

```
plot(fitted(mod_dl), resid_dl,
     main = "Residuen vs. Geschätzte Werte",
     xlab = "Geschätzte Werte (s)", ylab = "Residuen (s)")
abline(h = 0, lty = 2)
```



Hier schaut man auf Abweichungen relativ zum Modell–Output, nicht zur Variablen.

4.9 Schritt 9: Datenmenge in 15 Sekunden

Invertieren des Modells:

$$size = \frac{time - b_0}{b_1}$$

```
pred_size_15 <- (15 - coef(mod_dl)["(Intercept)"]) /
                coef(mod_dl)["size_mb"]
cat("In 15 s übertragbar:", round(pred_size_15,2), "MB\n")
```

```
## In 15 s übertragbar: 24.66 MB
```

Hinweis: Modell ist nur innerhalb des beobachteten Bereichs zuverlässig. Für grosse Extrapolationen ist ein anderes Modell (z. B. nicht-linear) empfehlenswert.

5 Aufgabe 4: Cars – Displacement vs. Horsepower

Aufgabenstellung: Im Datensatz **Cars** finden Sie Motor-Hubraum (`displacement`) und Leistung (`horsepower`).

1. Streudiagramm;
2. Lineares Modell;
3. Interpretation von R^2 & RSE;
4. Mehrleistung für +0.5 L?;
5. Residuum für 3 L/333 PS;
6. Beschreibung +/– Residuen;
7. RSE als Residuen-SD?;

8. 95 % CI für mittlere Leistung bei 3 L;
9. Wiederholung für 2 L und 6.2 L.

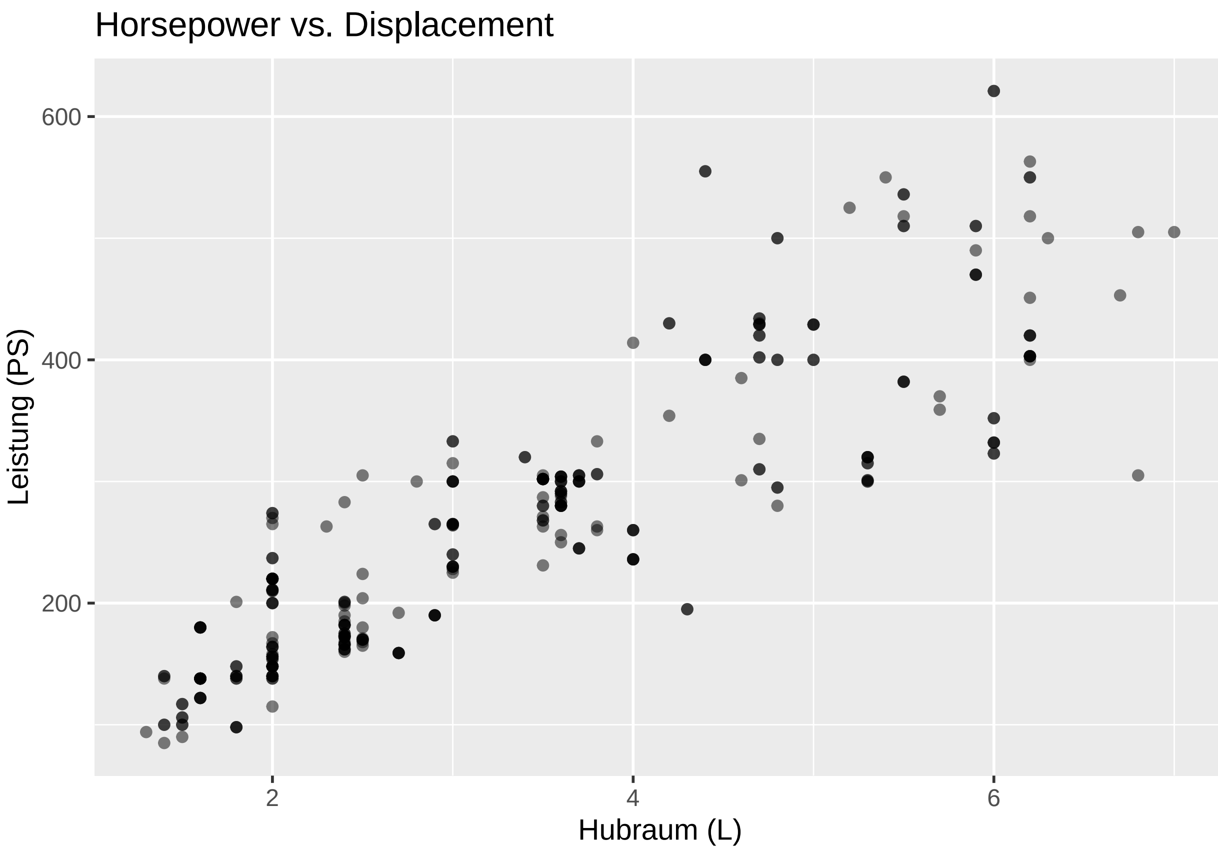
5.1 Schritt 1: Daten einlesen

```
cars <- read_excel("../data/WDDA_05.xlsx", sheet = "Cars") %>%
  rename(displacement = `Displacement (liters)`,
         horsepower   = Horsepower)
head(cars)
```

```
## # A tibble: 6 x 34
##   Name            Class    `Vehicle Type` Transmission `Combined MPG` `City MPG`
##   <chr>           <chr>    <chr>          <chr>                 <dbl>      <dbl>
## 1 Audi A3         Small S~ Car            Manual                   24         21
## 2 Audi A4 QUATTRO Compact~ Car            Manual                   25         21
## 3 Audi A4         Compact~ Car            Continuousl~             25         22
## 4 Audi A6         Midsize~ Both           Continuousl~             28         25
## 5 Audi Q7         Special~ Truck          Semi-Automa~             18         16
## 6 Audi R8         Two Sea~ Car            Automated M~             16         13
## # i 28 more variables: `Highway MPG` <dbl>, `Weight (pounds)` <dbl>,
## #   displacement <dbl>, horsepower <dbl>, `Air Aspiration` <chr>,
## #   `# Cylinders` <dbl>, `Intake Valves Per Cyl` <dbl>,
## #   `Exhaust Valves Per Cyl` <dbl>, `# Gears` <dbl>, `Drive System` <chr>,
## #   THC <dbl>, CO <dbl>, CO2 <dbl>, NOx <dbl>, PM <dbl>,
## #   `Energy Storage Device Desc` <chr>, `2Dr Pass Vol` <dbl>,
## #   `2Dr Lugg Vol` <dbl>, `4Dr Pass Vol` <dbl>, `4Dr Lugg Vol` <dbl>, ...
```

5.2 Schritt 2: Streudiagramm

```
ggplot(cars, aes(x = displacement, y = horsepower)) +
  geom_point(alpha = 0.5) +
  labs(title = "Horsepower vs. Displacement",
       x = "Hubraum (L)", y = "Leistung (PS)")
```



5.3 Schritt 3: Lineares Modell

```
mod_cars <- lm(horsepower ~ displacement, data = cars)
summary(mod_cars)
```

```
##
## Call:
## lm(formula = horsepower ~ displacement, data = cars)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -199.729  -34.263   -6.906   33.374  216.343
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    34.191      8.168   4.186 3.68e-05 ***
## displacement   69.197      2.192  31.564  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 57.72 on 316 degrees of freedom
## Multiple R-squared:  0.7592, Adjusted R-squared:  0.7584
## F-statistic: 996.3 on 1 and 316 DF,  p-value: < 2.2e-16
```

- **Intercept** $\approx$ 34.19 PS: Grundleistung bei 0 L (nicht realistisch).
- **Steigung** $\approx$ 69.20 PS/L: zusätzliche Leistung pro Liter.

5.4 Schritt 4: R^2 & RSE

```
resid_c <- resid(mod_cars)
rss_c   <- sum(resid_c^2)
tss_c   <- sum((cars$horsepower - mean(cars$horsepower))^2)
rse_c   <- sqrt(rss_c / df.residual(mod_cars))
r2_c    <- summary(mod_cars)$r.squared

cat("R^2 =", round(r2_c,4), "\n")
```

```
## R^2 = 0.7592
```

```
cat("RSE =", round(rse_c,2), "PS\n")
```

```
## RSE = 57.72 PS
```

5.5 Schritt 5: Mehrleistung für +0.5 L

$$\Delta \widehat{hp} = b_1 \times 0.5$$

```
delta_hp <- coef(mod_cars)["displacement"] * 0.5
cat("Geschätzte Mehrleistung:", round(delta_hp,1), "PS\n")
```

```
## Geschätzte Mehrleistung: 34.6 PS
```

> **Achtung:** Korrelation   Kausalität, aber für lineare Approximation kann man so vorgehen.

5.6 Schritt 6: Residuum für 3 L/333 PS

```
pred_3L <- predict(mod_cars, newdata = data.frame(displacement = 3))
resid_3L <- 333 - pred_3L
cat("Residual (333 PS bei 3 L):", round(resid_3L,2), "PS\n")
```

```
## Residual (333 PS bei 3 L): 91.22 PS
```

Da das Residuum **positiv** ist, liegt der Wagen **über** der Regressionsgerade (Performance-Fahrzeug).

5.7 Schritt 7: Beschreibung der Residuen

- **Positive Residuen:** Mehr Leistung als erwartet (z. B. Sportwagen).
- **Negative Residuen:** Weniger Leistung als erwartet (z. B. sparsamer Alltagswagen).

5.8 Schritt 8: RSE als SD der Residuen?

Per Definition ist RSE die Standardabweichung der Residuen – sinnvoll, solange keine starke Heteroskedastizität vorliegt.

5.9 Schritt 9: Konfidenzintervall für mittlere Leistung bei 3 L

```
cars3 <- filter(cars, displacement == 3)
t.test(cars3$horsepower)$conf.int
```

```
## [1] 251.4242 284.0044
## attr(,"conf.level")
## [1] 0.95
```

Sie erhalten ca. [**251, 284**] **PS**. Das Modell-Prediction ≈ 242 PS liegt teilweise ausserhalb – Hinweis auf Abweichung.

5.10 Schritt 10: Wiederholung für 2 L und 6.2 L

```
for(d in c(2, 6.2)) {
  subset <- filter(cars, displacement == d)
  ci     <- t.test(subset$horsepower)$conf.int
  cat("\nDisplacement =", d, "L:\n")
  cat("  95% CI:", round(ci[1],1), "-", round(ci[2],1), "PS\n")
  pred   <- predict(mod_cars, newdata = data.frame(displacement = d))
  cat("  Modell-Prediction:", round(pred,1), "PS\n")
}
```

```
##
## Displacement = 2 L:
##   95% CI: 172.8 - 198.5 PS
##   Modell-Prediction: 172.6 PS
##
## Displacement = 6.2 L:
##   95% CI: 413.4 - 487.6 PS
##   Modell-Prediction: 463.2 PS
```

Vergleichen Sie diese Intervalle mit den Modellvorhersagen und der globalen RSE.