Session 07:
Multiple Lineare Regression

Dominic Schmitz & Janina Esser

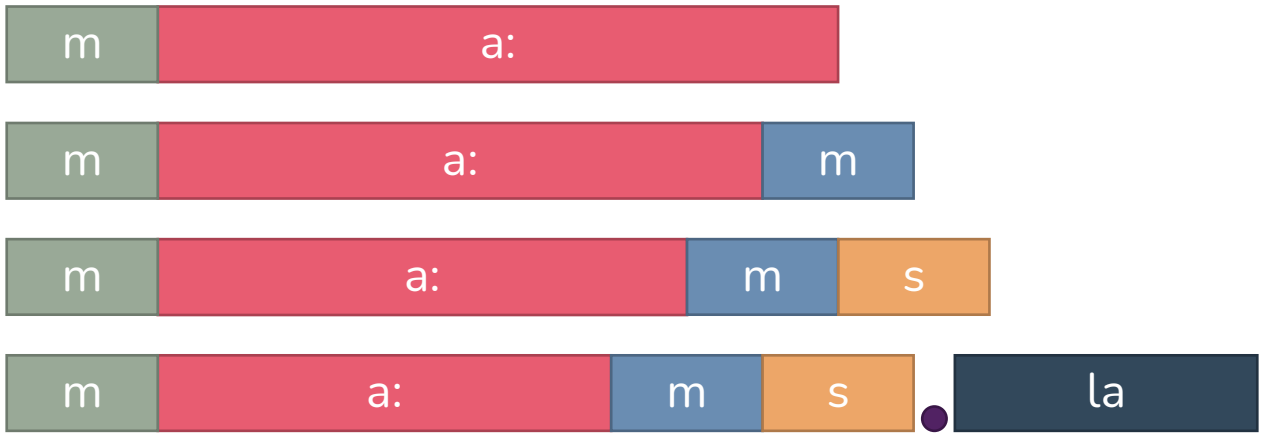
Verein für Diversität in der Linguistik

Beispieldaten

- Für die folgenden Beispiele werden wir Daten folgender Studie nutzen:

Compensatory Vowel Shortening in German[1]

- Stressed Vowels sind kürzer je nachdem wie viele Konsonanten ihnen folgen:



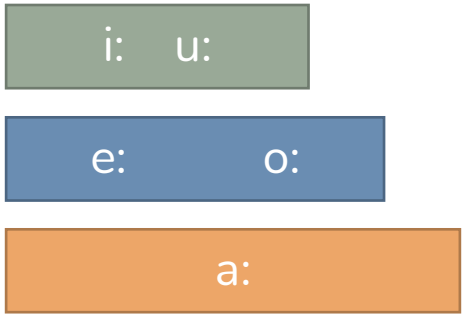
[1] Schmitz, D., Cho, H.-E., & Niemann, H. (2018). Vowel shortening in German as a function of syllable structure. Proceedings 13. Phonetik Und Phonologie Tagung (P&P13), 181–184.

Beispieldaten

- Für die folgenden Beispiele werden wir Daten folgender Studie nutzen:

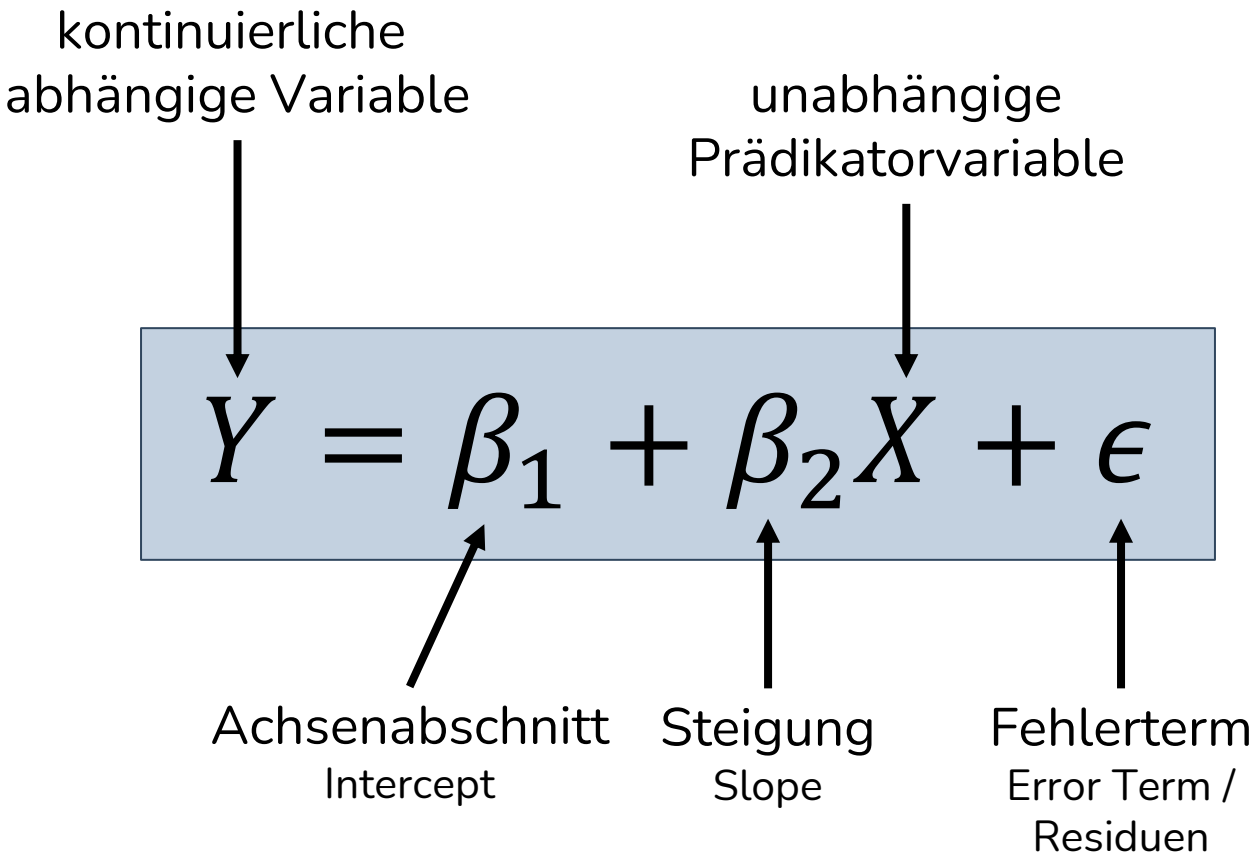
Compensatory Vowel Shortening in German[1]

- Unabhängig von diesem Vowel Shortening gilt, dass offene Vokale länger sind als halb-offene Vokale, und halb-offene Vokale sind länger als geschlossene Vokale:

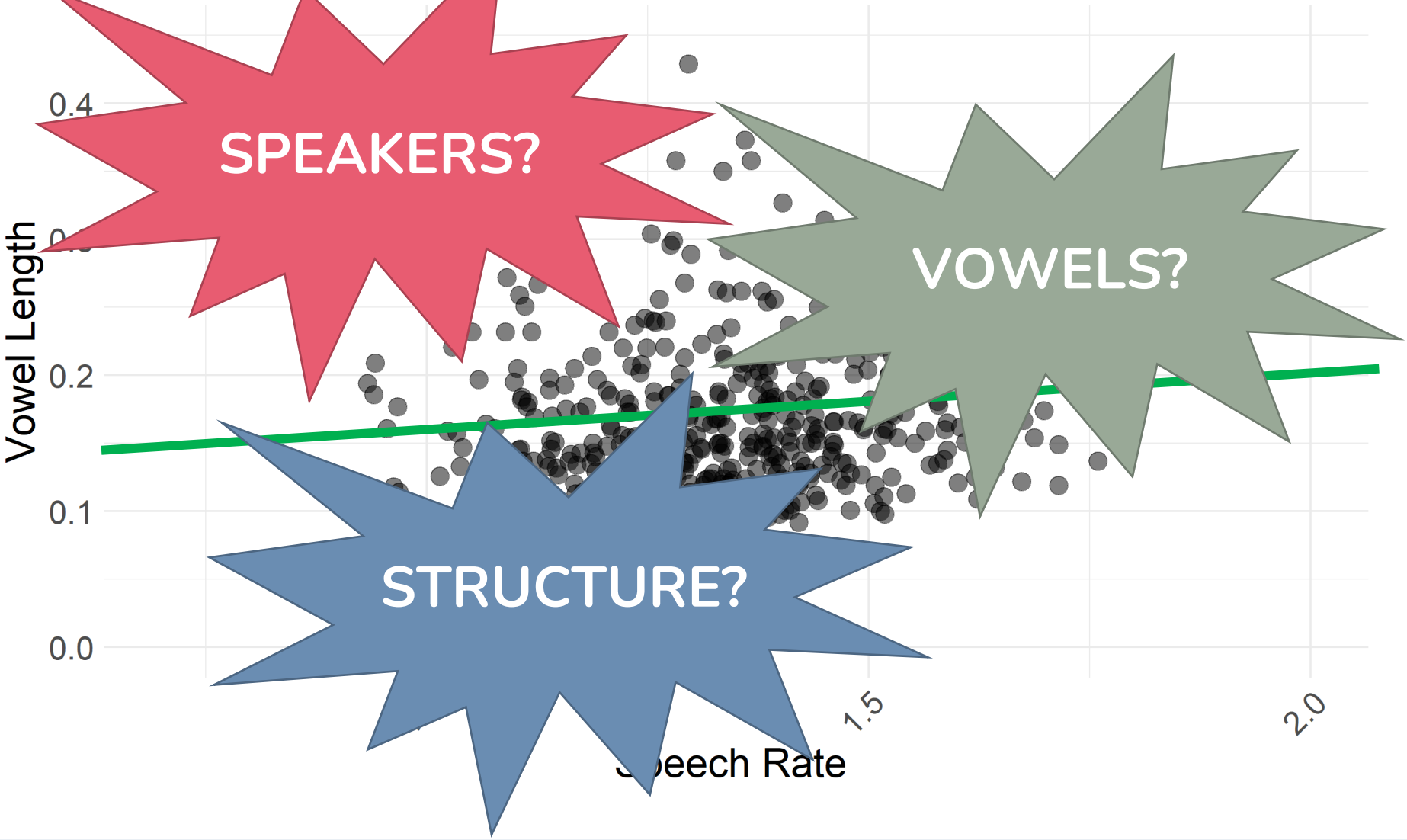


[1] Schmitz, D., Cho, H.-E., & Niemann, H. (2018). Vowel shortening in German as a function of syllable structure. Proceedings 13. Phonetik Und Phonologie Tagung (P&P13), 181–184.

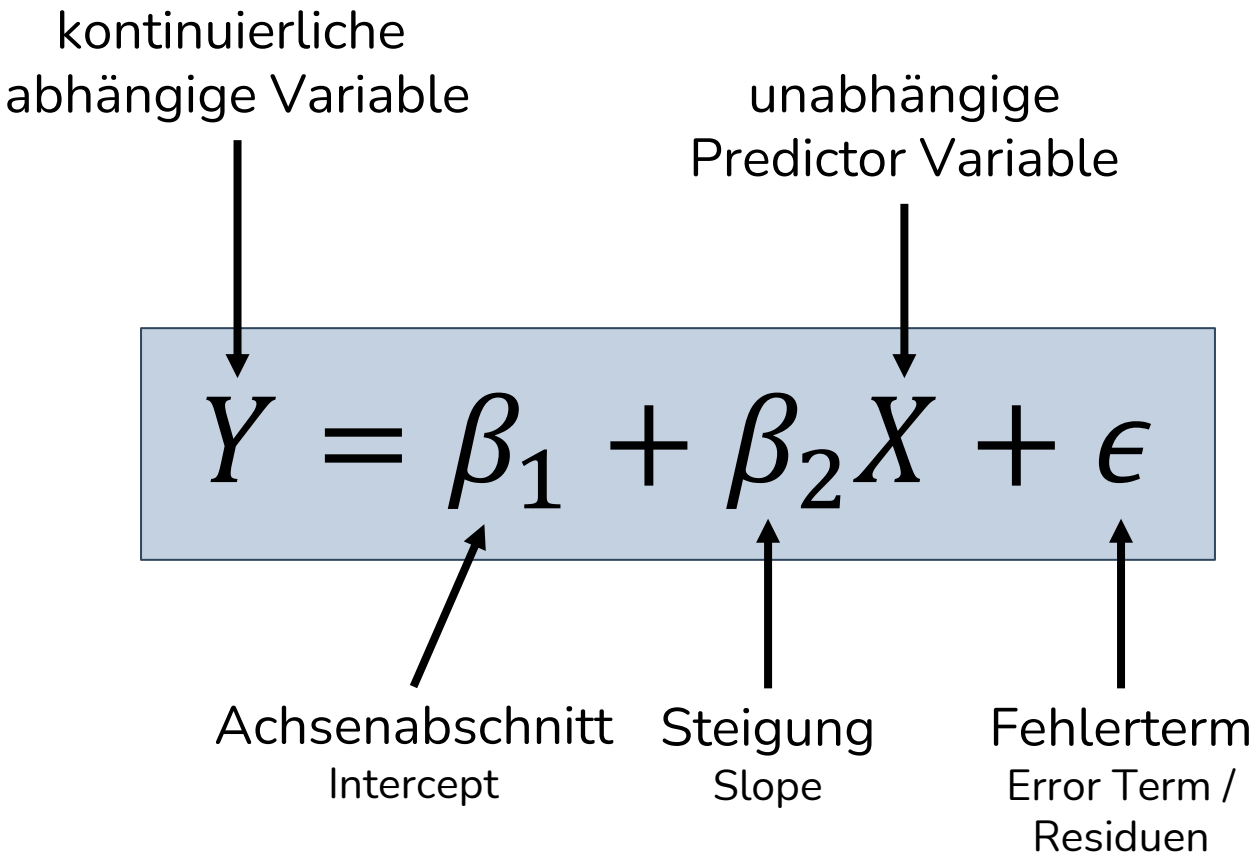
Einfache Lineare Regression



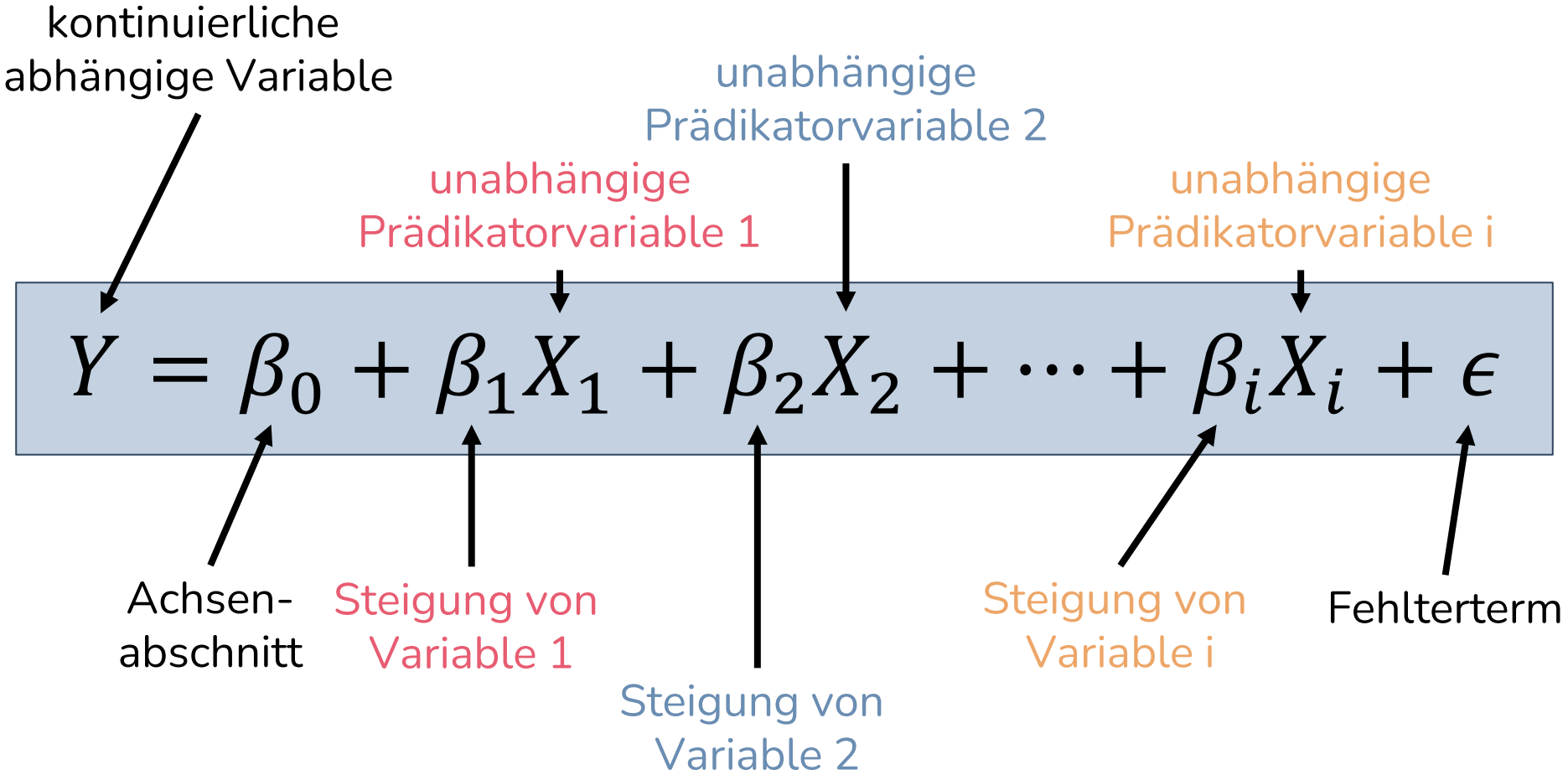
(Zu) Einfache Lineare Regression



Einfache Lineare Regression: Formel



Multiple Lineare Regression: Formel



Multiple Lineare Regression in R

- Mehr Variablen = mehr Zeitaufwand
- Typische Schritte bei Multipler Linearer Regression sind
  1. Distribution der abhängigen Variable überprüfen
  2. „volles“ Modell erstellen
  3. „bestes“ Modell finden
  4. Assumptions überprüfen
  5. Modell interpretieren

1: Verteilung der abhängigen Variable

- Wie wir bereits wissen, nutzen wir hierzu den Shapiro-Wilk Test
- Die abhängige Variable in unserem Beispiel, duration, ist nicht normalverteilt
- Daher nutzen wir wieder eine log-transformierte Version der Variable, `durationLog`

2: „Volles“ Modell

- Unsere abhängige Variable ist `durationLog`
- Als nächstes müssen wir die unabhängigen Variablen identifizieren, die wir nutzen möchten
- In diesem Beispiel sind es die folgenden Variablen:
  - `structure` i.e. coda structure
  - `vowel` i.e. vowel quality
  - `rate` i.e. speech rate
  - `number` i.e. slide number during experiment

2: „Volles“ Modell

- Erstellen des „vollen“ Modells:

```
model = lm(durationLog ~ structure +
                         vowel +
                         rate +
                         number,
                       data)
```

3: „Bestes“ Modell

- Theoretisch müssten wir nun alle möglichen Variabel-Kombinationen testen um das “beste” Modell zu finden
- Allerdings ist dieser Vorgang manuell durchgeführt fehleranfällig und zeitaufwendig (und macht wirklich keinen Spaß)
- Zum Glück gibt es eine Funktion, die diesen Schritt übernimmt:

```
step(model)
```

3: „Bestes“ Modell

```
> step(model)
```

3: „Bestes“ Modell

```
> step(model)

Start:  AIC=-1167.31

durationLog ~ structure + vowel + rate + number
```

Akaike Information Criterion
Je niedriger, desto besser die Modellanpassung

3: „Bestes“ Modell

```
> step(model)

Start:  AIC=-1167.31

durationLog ~ structure + vowel + rate + number
```

Akaike Information Criterion
Je niedriger, desto besser die Modellanpassung

	Df	Sum of Sq	RSS	AIC
- number	1	0.0536	31.839	-1168.55
<none>			31.786	-1167.31
- rate	1	0.8500	32.636	-1157.48
- vowel	4	3.4109	35.197	-1129.64
- structure	2	14.9708	46.756	-998.41

ein Modell ohne **number**

ein Modell ohne **vowel**

3: „Bestes“ Modell

```
Step:  AIC=-1168.55
durationLog ~ structure + vowel + rate
```

bestes Modell gefunden durch die `step()` function und sein AIC

```
             Df Sum of Sq    RSS      AIC
<none>                    31.839 -1168.55
- rate        1    0.8416 32.681 -1158.86
- vowel       4    3.4070 35.246 -1131.01
- structure   2   14.9881 46.827  -999.73
```

zusätzlicher Beweis dafür, dass eine weitere Reduzierung die Anpassung des Modells nicht verbessert

3: „Bestes“ Modell

bestes Modell gefunden durch die `step()` Funktion und ihren Aufruf

```
Call:
lm(formula = durationLog ~ structure + vowel + rate, data = data)

Coefficients:
    (Intercept)    structureopen  structuresingle           vowele
        -1.5062           0.4340           0.1219          -0.1441
         voweli           vowelo           vowelu             rate
        -0.2374          -0.1229          -0.2365          -0.2532
```

Modellkoeffizienten – werden wir uns in Schritt 5 genauer ansehen

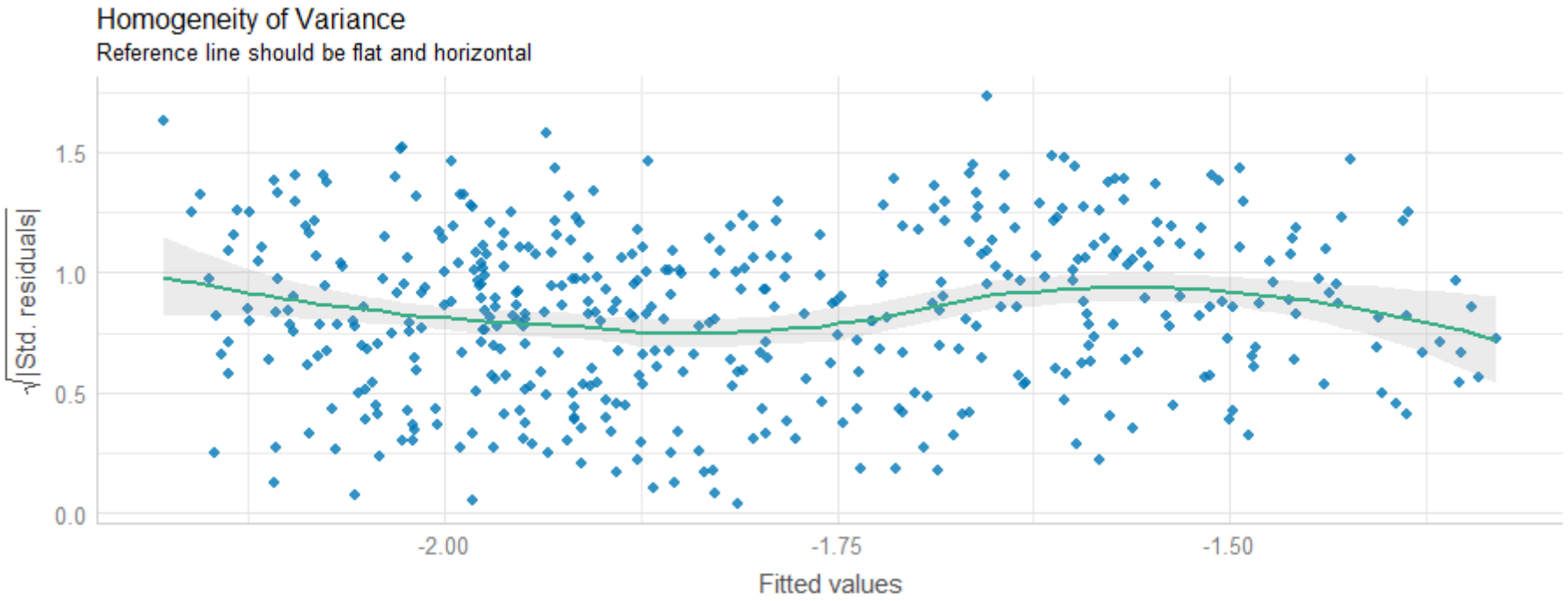
4: Annahmen überprüfen

- Multiple Lineare Regression folgt den gleichen Annahmen, denen auch die Einfache Lineare Regression folgt
  - Linearität
  - Homoskedastizität
  - Normalität
  - Unabhängigkeit

4: Annahmen überprüfen

- Linearitätsannahme:

  Die Beziehung zwischen X und dem Mittelwert von Y ist linear.

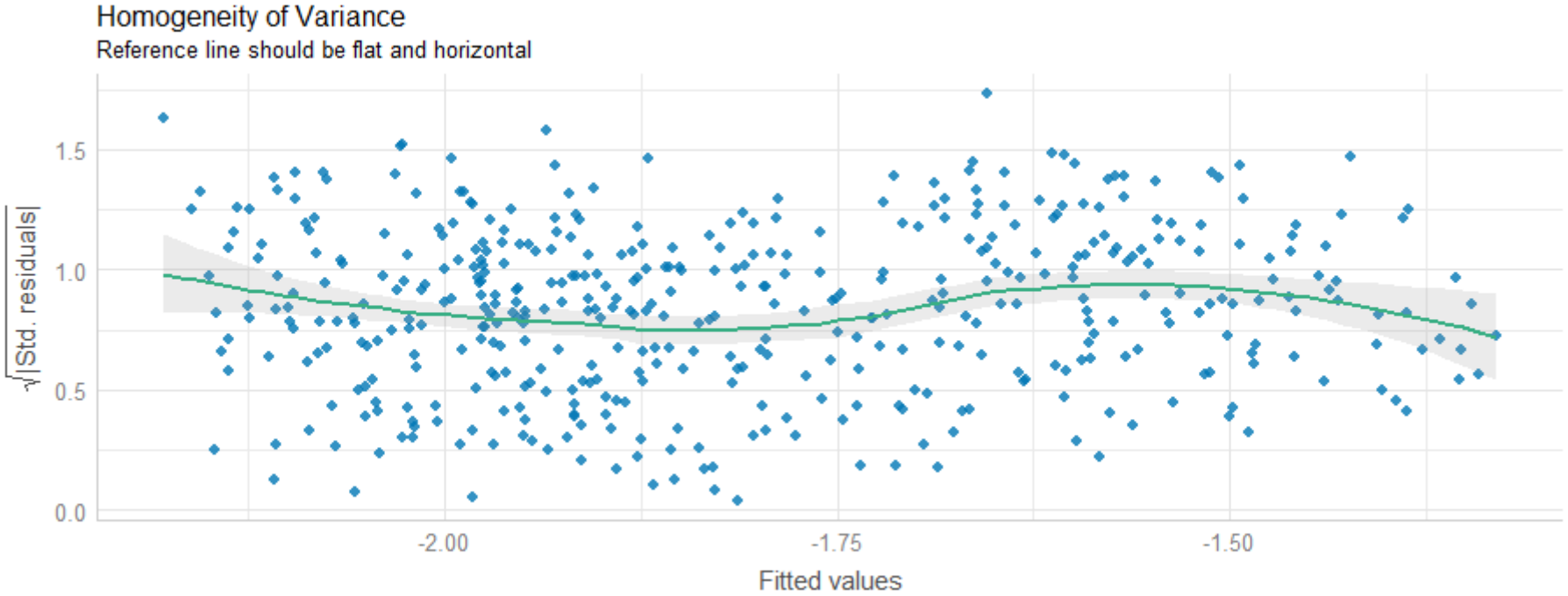


- Die Linie sollte horizontal und flach sein.

4: Annahmen überprüfen

- Homoskedastizitätsannahme:

  Die Varianz der Residuen ist für jeden Wert von X gleich groß.

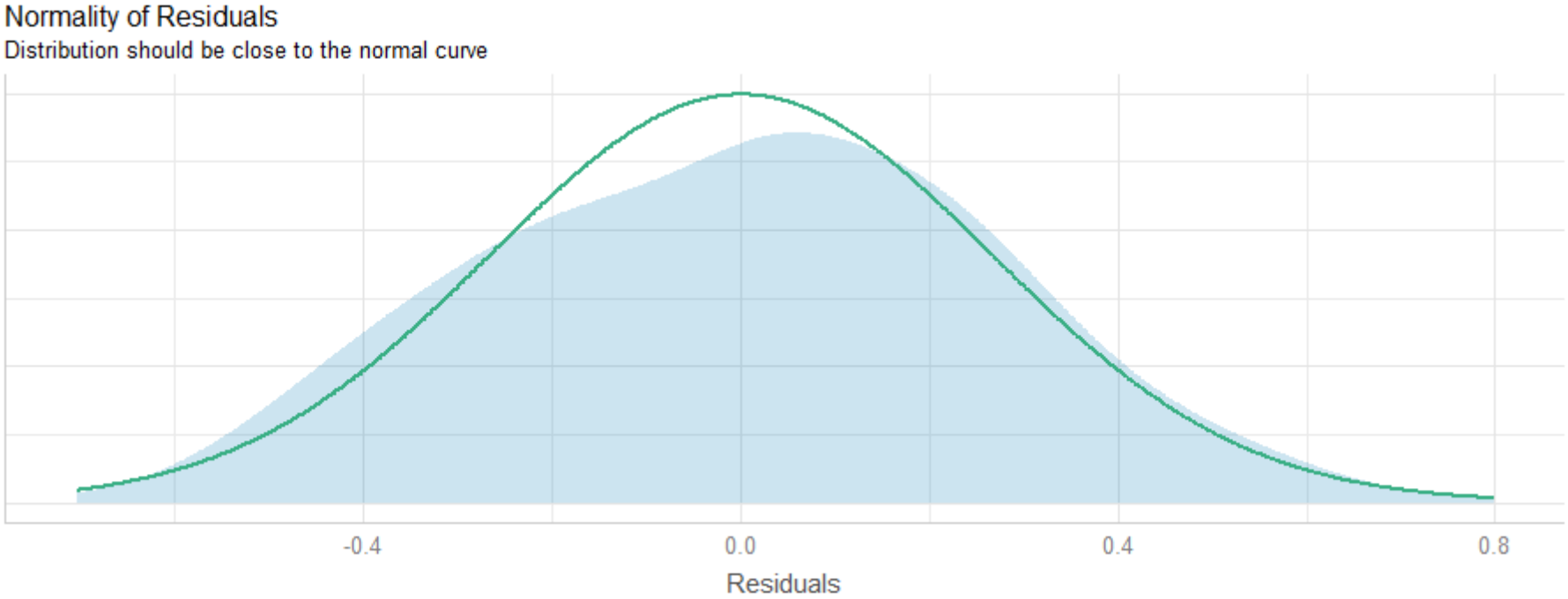


- Die Daten sollten gleichmäßig über die Linie verteilt sein, wobei keine offensichtlichen Muster erkennbar sein sollten.

4: Annahmen überprüfen

- Normalitätsannahme:

  Für jeden festen Wert von X ist Y normalverteilt.



- Die Verteilung der Residuen eines linearen Modells sollte einer Normalverteilung folgen.

5: Interpretation

- Generell sind wir an zwei Dingen interessiert:
    1. die *p*-Werte der einzelnen Prädikatoren / Predictors
    2. die Effekte der einzelnen Prädikatoren / Predictors

5: Interpretation – *p*-Werte

1. Mit der `anova()` Funktion erhalten wir *p*-Werte

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
structure	2	15.131	7.5654	104.4874	< 2.2e-16	***
vowel	4	3.507	0.8767	12.1079	2.41e-09	***
rate	1	0.842	0.8416	11.6241	0.0007112	***
Residuals	439	31.786	0.0724			

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

```
                Estimate   Std. Error  t value  Pr(>|t|)
(Intercept)     -1.50620     0.10486  -14.364   < 2e-16 ***
structureopen    0.43395     0.03112   13.947   < 2e-16 ***
structuresingle  0.12186     0.03117    3.910  0.000107 ***
vowele          -0.14406     0.04033   -3.572  0.000393 ***
voweli          -0.23739     0.04035   -5.883  7.97e-09 ***
vowelo          -0.12292     0.04034   -3.048  0.002446 **
vowelu          -0.23653     0.04033   -5.864  8.87e-09 ***
rate            -0.25324     0.07425   -3.410  0.000708 ***
```

Step 5: Interpretation – Effects

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.04513	13.947	< 2e-16	***
structuresingle	0				***
vowele	-0				***
voweli	-0				***
vowelo	-0				**
vowelu	-0				***
rate	-0				***

enthält die Ausgangsniveaus aller Faktoren, d. h.

structure:double
vowel:a

sowie der "Ausgangspunkt" des/der numerischen Prädiktoren

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Predictors werfen

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.03112	13.947	< 2e-16	***
structuresingle	0.[illegible]				***
vowele	-0.[illegible]				***
voweli	-0.23739	0.04035	-5.883	7.97e-09	***
vowelo	-0.12292	0.04034	-3.048	0.002446	**
vowelu	-0.23653	0.04033	-5.864	8.87e-09	***
rate	-0.25324	0.07425	-3.410	0.000708	***

structure:double + vowel:a + rate:start

geschätzter Mittelwert von `durationLog`

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Predictors werfen

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.03112	13.947	< 2e-16	***
structuresingle	0.[illegible]				***
vowele	-0.[illegible]				***
voweli	-0.23739	0.04035	-5.883	7.97e-09	***
vowelo	-0.12292	0.04034	-3.048	0.002446	**
vowelu	-0.23653	0.04033	-5.864	8.87e-09	***
rate	-0.25324	0.07425	-3.410	0.000708	***

structure:double + vowel:a + rate:start

Standardfehler dieses Mittelwerts

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Predictors werfen

structure:double + vowel:a + rate:start

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.03112	13.947	< 2e-16	***
structuresingle	0.12186	0.03117	3.910	0.000107	***
vowele					
voweli					
vowelo					
vowelu					
rate					

Um den geschätzten Mittelwert von `durationLog` in structure:single words zu erhalten, müssen wir seine Schätzung zum Achsenabschnitt addieren, d. h.

-1.50620 + 0.12186 = -1.38434

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

structure:single + vowel:a + rate:start

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.03112	13.947	< 2e-16	***
structuresingle	0.12186	0.03117	3.910	0.000107	***
vowele	-0.14406	0.04033	-3.572	0.000393	***
voweli	-0.23739	0.04035	-5.883	7.97e-09	***
vowelo					
vowelu					
rate					

Um den geschätzten Mittelwert von `durationLog` in structure:single & vowel:i-Wörtern zu erhalten, müssen wir beide Schätzungen zum Intercept addieren, d. h.

-1.50620 + 0.12186 - 0.23739 = -1.62173

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Predictors werfen

structure:double + vowel:a + rate:start

	Estimate			
(Intercept)	-1.50620			***
structureopen	0.43395			***
structuresingle	0.12186			***
vowele	-0.14406			***
voweli	-0.23739			***
vowelo	-0.12292			**
vowelu	-0.23653	0.04033	-5.864	8.87e-09 ***
rate	-0.25324	0.07425	-3.410	0.000708 ***

`durationLog` is

- deutlich länger in offenen coda-Wörtern
- deutlich länger bei einfachen coda-Wörtern

als in komplexen coda-Wörtern

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

structure:double + vowel:a + rate:start

```
                 Estimate  Std. Error  t value  Pr(>|t|)
(Intercept)      -1.50620     0.10486  -14.364   < 2e-16 ***
structureopen     0.43395     0.03112   13.947   < 2e-16 ***
structuresingle   0.12186     0.03117    3.910  0.000107 ***
vowele           -0.14406                                ***
voweli           -0.23739                                ***
vowelo           -0.12292                                **
vowelu           -0.23653                                ***
rate             -0.25324                                ***
```

`durationLog` ist

- deutlich kürzer in Wörtern mit allen anderen Vokalen, d.h. /e, i, o, u/

als in Wörtern mit /a/

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	-1.50620	0.10486	-14.364	< 2e-16	***
structureopen	0.43395	0.03112	13.947	< 2e-16	***
structuresingle	0.12186	0.03117	3.910	0.000107	***
vowele	-0.14406	0.04033	-3.572	0.000393	***
voweli				7.97e-09	***
vowelo				0.002446	**
vowelu				8.87e-09	***
rate	-0.25324	0.07425	-3.410	0.000708	***

je höher die Sprechrate, desto niedriger ist der Wert von `durationLog`

Step 5: Interpretation – Effekte

2. Mit der `summary()` Funktion können wir einen Blick auf die einzelnen Effekte der Prädikatoren werfen

```
                Estimate   Std. Error  t value  Pr(>|t|)
(Intercept)     -1.50620     0.10486  -14.364   < 2e-16 ***
structureopen    0.43395     0.03112   13.947   < 2e-16 ***
structuresingle  0.12186     0.03117    3.910  0.000107 ***
vowele          -0.14406     0.04033   -3.572  0.000393 ***
voweli          -0.23739     0.04035   -5.883  7.97e-09 ***
vowelo          -0.12292     0.04034   -3.048  0.002446 **
vowelu          -0.23653     0.04033   -5.864  8.87e-09 ***
rate            -0.25324     0.07425   -3.410  0.000708 ***
```

Step 5: Interpretation – Effekte

Der s.g. „Tukey-Contrast“ zeigt uns die Unterschiede innerhalb eines kategorischen Prädikators

```
> tukey(model = mdl_fin, predictor = structure)

                    Estimate Std. Error t value Pr(>|t|)
open - double == 0   0.43395    0.03112   13.95  < 1e-04 ***
single - double == 0 0.12186    0.03117    3.91  0.00031 ***
single - open == 0  -0.31209    0.03111  -10.03  < 1e-04 ***
```