Linear Models in R (M1–MIDO)

Lab Session 2 — Solutions

Henri PANJO

Table of contents

Dataset Overview: *data_pokemon.csv*

This dataset is adapted from a popular Kaggle Pokémon dataset.
Even if you are not familiar with Pokémon, the data is straightforward:
it combines numeric statistics with categorical attributes, making it well-suited for applying **Ordinary Least Squares (OLS)** in R.

What it contains

- Unique identifiers and names for each Pokémon
- Battle statistics (health, attack, defense, special attack, special defense, speed)
- Categorical features (primary/secondary type, generation, legendary flag)

Fields (Codebook)

- `id`: Unique Pokémon ID
- `name`: Pokémon name
- `type_1`: Primary type (e.g., Water, Fire)
- `type_2`: Secondary type (optional)
- `hp`: Hit points (overall health)
- `attack`: Physical attack strength (**we will use this as y** in most regressions)
- `defense`: Physical defense strength
- `sp_attack`: Special (non-physical) attack strength
- `sp_defense`: Special defense strength
- `speed`: Speed / turn order
- `generation`: Game generation label
- `legendary`: Indicator for legendary status (TRUE/FALSE)

Note on notation

- We treat `attack` as the outcome variable Y.
- Predictor variables (e.g., `defense`, `speed`) will be denoted as $x_1, x_2, \ldots$.
- Factors like `type_1` or `legendary` will be included as categorical predictors.

Setup

To keep numbers readable and reproducible, we set display options:

```r
options(scipen = 999, digits = 5)
```

We also load the packages used during this session.

> **Warning**
>
> Don't worry if you don't know them all — we'll introduce functions as we need them. Some provide regression tools, others are for data visualization or diagnostics.

```r
library(broom)
library(performance)
library(parameters)
library(datawizard)
library(see)
library(effectsize)
library(insight)
library(correlation)
library(modelbased)
library(glue)
library(scales)
library(GGally)
library(ggpubr)
library(car)
library(lmtest)
library(rstatix)
library(matrixTests)
library(ggfortify)
library(qqplotr)
library(patchwork)
library(gtsummary)
library(kableExtra)
library(collapse)
library(tidyverse)
```

```r
source("helper_functions.R")
```

Question 1. Loading dataset

Import the dataset `data_pokemon.csv` with `read_csv()` and save it in an object called `pok`.
Using `select()`, keep only the variables `id`, `name`, `attack`, `speed`, `defense`, `hp`, `sp_attack`, and `sp_def`.

- Display the first 10 rows of `pok` using `head()` or `slice()`.

Solutions

- Loading `data_pokemon.csv`

```
pok <- read_csv("data_pokemon.csv", show_col_types = FALSE) |>
  select(id, name, attack, speed, defense, hp, sp_attack, sp_def)
```

- `head()` on `pok`

```
head(pok, n = 10)
```

```
# A tibble: 10 x 8
      id name              attack speed defense    hp sp_attack sp_def
   <dbl> <chr>              <dbl> <dbl>   <dbl> <dbl>     <dbl>  <dbl>
 1     1 Bulbasaur             49    45      49    45        65     65
 2     2 Ivysaur               62    60      63    60        80     80
 3     3 Venusaur              82    80      83    80       100    100
 4     4 Mega Venusaur        100    80     123    80       122    120
 5     5 Charmander            52    65      43    39        60     50
 6     6 Charmeleon            64    80      58    58        80     65
 7     7 Charizard             84   100      78    78       109     85
 8     8 Mega Charizard X     130   100     111    78       130     85
 9     9 Mega Charizard Y     104   100      78    78       159    115
10    10 Squirtle              48    43      65    44        50     64
```

Question 2. Data management, label variable

Attach descriptive labels to each variable in the dataset `pok`.
This helps make outputs (e.g., summaries or regression tables) more readable.

Hint: use `relabel()` from the `{collapse}` package.

Solutions

- **Add labels to each variable** : we use `relabel()` to attach human-friendly names to our variables.

```r
pok <- pok |>
  relabel(
    attack = "Attack power", speed = "Speed power", defense = "Defense power",
    hp = "Hit points (health)", sp_attack = "Special attack power",
    sp_def = "Special defense power", id = "ID", name = "Pokemon name"
  )
```

- **Check that labels were added:** the function `namlab()` (from `{collapse}`) shows variable names, labels, and basic info.

```r
namlab(pok, N = TRUE, Ndistinct = TRUE, class = TRUE)
```

```
   Variable     Class   N Ndist                 Label
1        id   numeric 800   800                    ID
2      name character 800   800          Pokemon name
3    attack   numeric 800   111          Attack power
4     speed   numeric 800   108           Speed power
5   defense   numeric 800   103         Defense power
6        hp   numeric 800    94   Hit points (health)
7 sp_attack   numeric 800   105  Special attack power
8    sp_def   numeric 800    92 Special defense power
```

- **To check only for some variables**, we can use `vlabels()` from `{collapse}`

```r
vlabels(pok$attack)
```

```
[1] "Attack power"
```

```r
vlabels(pok[c("speed", "defense")])
```

```
          speed         defense
  "Speed power" "Defense power"
```

Question 3. Summary statistics

For the variables `attack`, `speed`, `defense`, `hp`, `sp_attack`, `sp_def`, compute summary statistics: mean, standard deviation, median, Q1, Q3, minimum, maximum.

Hint: `descr()`, `describe_distribution()`, `get_summary_stats()` `summarise()`, `tbl_summary()`

Solutions

- **Save numeric variable names in** `numeric_vars`

```r
numeric_vars <- names(pok)[-c(1, 2)]
numeric_vars
```

```
[1] "attack"    "speed"     "defense"   "hp"        "sp_attack" "sp_def"
```

- **Save predictors variable names in** `predictors`

```r
predictors <- numeric_vars[-1]
predictors
```

```
[1] "speed"     "defense"   "hp"        "sp_attack" "sp_def"
```

- Generate summary statistics with `tbl_summary()` from `{gtsummary}`

```
tab_summary <- select(pok, all_of(numeric_vars)) |>
  tbl_summary(
    type = all_continuous() ~ "continuous2",
    statistic = all_continuous() ~ c(
      "{mean} ({sd})", "{median} ({p25}, {p75})", "{min}, {max}"
    ),
    digits = ~ 1
  ) |>
  bold_labels()
```

- Display the table in the PDF using `{kableExtra}`

```
as_kable_extra(tab_summary, booktabs = TRUE, longtable = TRUE, linesep = "") |>
  kable_styling(
    position = "center", font_size = 10,
    latex_options = c("basic", "repeat_header")
  )
```

Characteristic	N = 800
Attack power	
Mean (SD)	79.0 (32.5)
Median (Q1, Q3)	75.0 (55.0, 100.0)
Min, Max	5.0, 190.0
Speed power	
Mean (SD)	68.3 (29.1)
Median (Q1, Q3)	65.0 (45.0, 90.0)
Min, Max	5.0, 180.0
Defense power	
Mean (SD)	73.8 (31.2)
Median (Q1, Q3)	70.0 (50.0, 90.0)
Min, Max	5.0, 230.0
Hit points (health)	
Mean (SD)	69.3 (25.5)
Median (Q1, Q3)	65.0 (50.0, 80.0)
Min, Max	1.0, 255.0
Special attack power	
Mean (SD)	72.8 (32.7)
Median (Q1, Q3)	65.0 (49.5, 95.0)
Min, Max	10.0, 194.0
Special defense power	
Mean (SD)	71.9 (27.8)
Median (Q1, Q3)	70.0 (50.0, 90.0)
Min, Max	20.0, 230.0

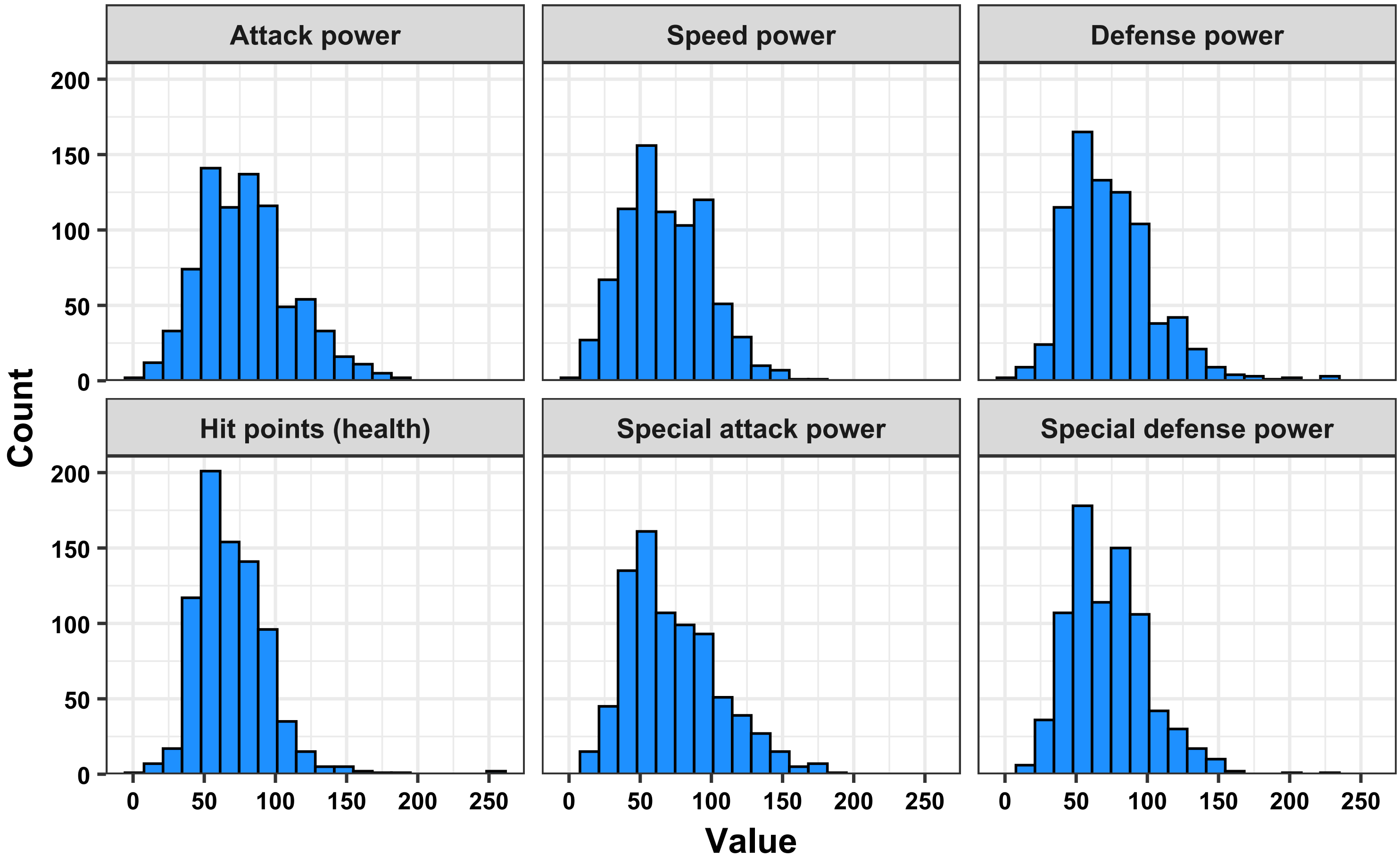
Question 4. Histogram, Scatter plots, Correlations

1. Plot histogram of `attack`, `speed`, `defense`, `hp`, `sp_attack`, `sp_def`.
2. Create scatter plots of `attack`, `speed`, `defense`, `hp`, `sp_attack`, `sp_def` against each other using `ggpairs()` from `{GGally}`.

Solutions

- **Histogram of** `attack`, `speed`, `defense`, `hp`, `sp_attack`, `sp_def`

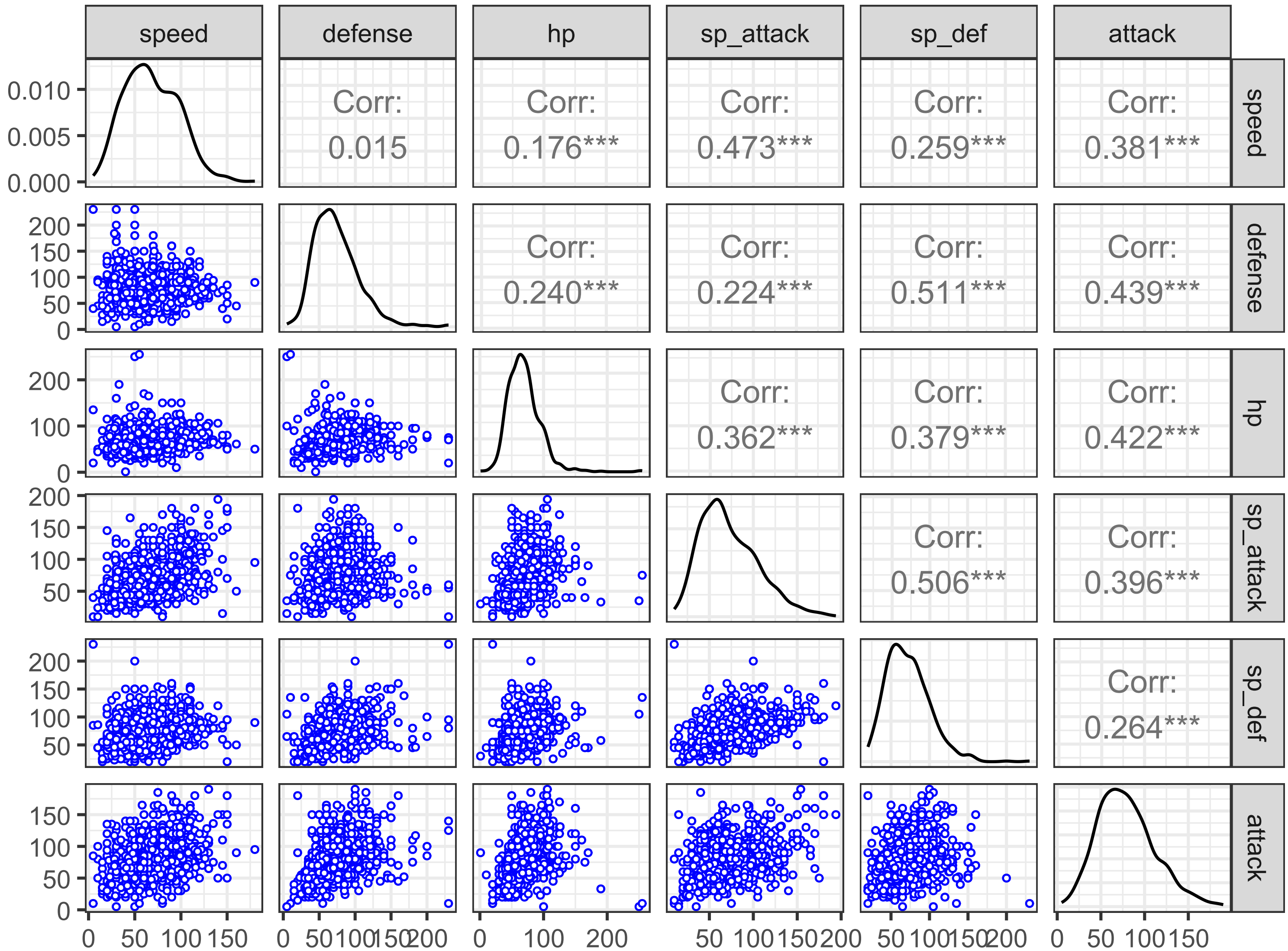
```
select(pok, id, attack:sp_def) |> pivot_longer(-id, names_to = "var") |>
  mutate(var = factor(var, levels = numeric_vars, labels = vlabels(pok[, numeric_vars]))) |>
  ggplot(aes(x = value)) +
  facet_wrap(vars(var)) +
  geom_histogram(fill = "dodgerblue", color = "black", bins = 20, linewidth = 0.5) +
  scale_y_continuous(expand = expansion(c(0, 0.05))) +
  scale_x_continuous(breaks = pretty_breaks()) +
  labs(x = "Value", y = "Count") + theme_bw(base_size = 14) +
  theme(strip.text = element_text(size = 11, face = "bold")) +
  labs_pubr()
```



Attack power
Speed power
Defense power
Hit points (health)
Special attack power
Special defense power
Count
Value
0
50
100
150
200
250

- Scatter plots of `attack`, `speed`, `defense`, `hp`, `sp_attack`, `sp_def` with `ggpairs()`

```
select(pok, -id, -name) |>
  relocate(attack, .after = last_col()) |>
  ggpairs(
    lower = list(
      continuous = wrap(
        "points",
        size = 1, shape = 21, fill = "white", color = "blue", alpha = 1
      )
    )
  ) +
  theme_bw(base_size = 14)
```



Question 5. Multivariate linear gaussian regression model

We now fit a multivariate linear model of `speed`, `defense`, `hp`, `sp_attack`, `sp_def` on `attack`.

Scalar form

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$
$$y_i = \mathbf{x}_i^\top \boldsymbol{\beta} + \varepsilon_i \qquad p = 5,\ i = 1, \cdots, n, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2).$$

with $y_i = \texttt{attack}_i$ and $(x_{i1}, \dots, x_{i5}) = (\texttt{speed}_i,\ \texttt{defense}_i,\ \texttt{hp}_i,\ \texttt{sp\_attack}_i,\ \texttt{sp\_def}_i)$

$$\texttt{attack} = \beta_0 + \beta_1 \texttt{speed} + \beta_2 \texttt{defense} + \beta_3 \texttt{hp} + \beta_4 \texttt{sp\_attack} + \beta_5 \texttt{sp\_def} + \varepsilon$$

Matrix form

$$\mathbf{y} = \mathbb{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}\left(\mathbf{0}_n, \sigma^2 \mathbf{I}_n\right), \mathrm{rank}(\mathbb{X}) = p + 1 = 6$$

1. Fit the model with `lm()` and save the result in `full_model`.
2. Interpret the output of:

```r
summary(full_model)
model_parameters(full_model, pretty_names = FALSE)
```

Solutions

- Generate the model formula with `reformulate()`

```r
full_formula <- reformulate(termlabels = predictors, response = "attack")
full_formula
```

```
attack ~ speed + defense + hp + sp_attack + sp_def
```

```r
class(full_formula)
```

```
[1] "formula"
```

- Fit the full model with `lm()` and save it in `full_model`

```r
full_model <- lm(full_formula, data = pok)
```

- **Examine the model output**

```r
summary(full_model)
```

```
Call:
lm(formula = full_formula, data = pok)

Residuals:
   Min     1Q Median     3Q    Max 
-86.93 -15.90  -2.48  13.50  95.15 

Coefficients:
            Estimate Std. Error t value            Pr(>|t|)    
(Intercept)   3.5187     3.3987    1.04                 0.3    
speed         0.3422     0.0340   10.07 < 0.0000000000000002 ***
defense       0.4677     0.0326   14.36 < 0.0000000000000002 ***
hp            0.3700     0.0374    9.88 < 0.0000000000000002 ***
sp_attack     0.1654     0.0342    4.84        0.000001554878 ***
sp_def       -0.2794     0.0416   -6.71        0.000000000037 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 24.4 on 794 degrees of freedom
Multiple R-squared:  0.44,  Adjusted R-squared:  0.436 
F-statistic:  125 on 5 and 794 DF,  p-value: <0.0000000000000002
```

$$\hat{\beta} = (\mathbb{X}^\top\mathbb{X})^{-1}\mathbb{X}^\top\mathbf{y} = (3.519, 0.342, 0.468, 0.370, 0.165, -0.279)^\top$$

```r
model_parameters(full_model, pretty_names = FALSE)
```

Parameter	Coefficient	SE	95% CI	t(794)	p
(Intercept)	3.519	3.399	[-3.153, 10.190]	1.035	0.301
speed	0.342	0.034	[0.275, 0.409]	10.067	< .001
defense	0.468	0.033	[0.404, 0.532]	14.361	< .001
hp	0.370	0.037	[0.297, 0.444]	9.883	< .001
sp_attack	0.165	0.034	[0.098, 0.232]	4.841	< .001
sp_def	-0.279	0.042	[-0.361, -0.198]	-6.711	< .001

$$\widehat{\texttt{attack}} = 3.519 + 0.342 \cdot \texttt{speed} + 0.468 \cdot \texttt{defense} + 0.37 \cdot \texttt{hp} + 0.165 \cdot \texttt{sp\_attack} - 0.279 \cdot \texttt{sp\_def}$$

$$\hat{\mathbf{y}} = \mathbb{X}\hat{\beta} = \mathbb{X}(\mathbb{X}^\top\mathbb{X})^{-1}\mathbb{X}^\top\mathbf{y} = P_{\mathbb{X}}\mathbf{y} = H_{\mathbb{X}}\mathbf{y}$$

Interpretation of the regression output

$$\texttt{attack}_i = \beta_0 + \beta_1 \texttt{speed}_i + \beta_2 \texttt{defense}_i + \beta_3 \texttt{hp}_i + \beta_4 \texttt{sp\_attack}_i + \beta_5 \texttt{sp\_def}_i + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2).$$

---

1. Model fit (global statistics)

$$\hat{\sigma}^2 = \frac{1}{n-r} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \frac{\hat{\varepsilon}^\top \hat{\varepsilon}}{n-r} = \frac{\|\hat{\varepsilon}\|^2}{n-r} = \frac{\text{RSS}}{n-r}$$

- **Residual standard error (estimated standard deviation of the errors):** $\hat{\sigma} = 24.4$
  On average, predictions of `attack` deviate from the true values by about 24 points.

$$R^2 = \frac{\text{MSS}}{\text{TSS}} = \frac{\sum_{i=1}^{n} (\hat{y}_i - \overline{y})^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2} = 1 - \frac{\text{RSS}}{\text{TSS}} = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2}$$

- $R^2 = 0.44$ (Adjusted $R^2 = 0.436$)
  The model explains about **44% of the variation** in attack strength across Pokémon.

- **F-statistic = 125 on (5, 794) df**, ($p < 2 \times 10^{-16}$)
  The model as a whole is highly significant; at least one predictor is associated with `attack`.

---

2. Coefficients

- We interpret coefficients while **holding other predictors constant**.

- **Intercept** ($\hat{\beta}_0 = 3.52, p = 0.30$) Not statistically significant. Represents expected attack when all predictors are 0 (not meaningful here, but needed for the model).

- **Speed** ($\hat{\beta}_1 = 0.34, p < 0.001$) A one-unit increase in speed is associated with a **0.34 increase in attack**. 95% CI: [0.28, 0.41].

- **Defense** ($\hat{\beta}_2 = 0.47, p < 0.001$) A one-unit increase in defense is associated with a **0.47 increase in attack**. Strongest positive effect, $CI : [0.40, 0.53]$.

- **HP** ($\hat{\beta}_3 = 0.37, p < 0.001$) A one-unit increase in HP is associated with a **0.37 increase in attack.** $CI : [0.30, 0.44]$.

- **Special Attack** ($\hat{\beta}_4 = 0.17, p < 0.001$) A one-unit increase in special attack is associated with a **0.17 increase in attack**. $CI : [0.10, 0.23]$.

- **Special Defense** ($\hat{\beta}_5 = -0.28, p < 0.001$) A one-unit increase in special defense is associated with a **0.28 decrease in attack**. $CI : [-0.36, -0.20]$.
  Suggests a tradeoff: high special defense Pokémon tend to have weaker physical attack.

3. Substantive interpretation

- Pokémon with higher **defense**, HP, and **speed** tend to also have higher attack.
- **Special attack** contributes positively, but less strongly.
- **Special defense** shows an *inverse relationship*: more defensive Pokémon are less offensively strong.

Conclusion:
The regression explains about **44% of the variance in attack strength**. Most predictors are significant and meaningful: defense, HP, speed, and special attack increase attack power, while special defense decreases it.

Question 6. Test of overall regression

We want to test

$$
\begin{aligned}
& H_0 : \beta_1 = \beta_2 = \cdots = \beta_5 = 0 \quad \text{versus} \quad H_1 : \text{At least one } \beta_j \neq 0 \\
\Longleftrightarrow & H_0 : (m_0)\ \mathbf{y} = \beta_0 \mathbf{1}_n + \boldsymbol{\varepsilon} \quad \text{versus} \quad H_1 : (m_1)\ \mathbf{y} = \mathbb{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}
\end{aligned}
$$

Perform this test in 2 differents ways. **Hint:** Fisher test for nested models / General linear hypothesis tests

Method 1: Fisher test for nested models
The F-statistic is

$$
F = \frac{\left\| P_{m_0}\mathbf{y} - P_{m_1}\mathbf{y} \right\|^2 / (p-q)}{\left\| \mathbf{y} - P_{m_1}\mathbf{y} \right\|^2 / (n-r)} = \frac{[\text{RSS}(m_0) - \text{RSS}(m_1)] / (p-q)}{\text{RSS}(m_1)/(n-r)} \sim F_{p-q,n-r} \quad (\text{under } H_0)
$$

- Here $q = 0$ (reduced model has only an intercept),
- $r = p + 1 = 6$,
- $p - q = 5$.

Method 2: General linear hypothesis test
We can also write

$$
\begin{aligned}
& H_0 : \beta_1 = \beta_2 = \cdots = \beta_5 = 0 \quad \text{versus} \quad H_1 : \text{At least one } \beta_j \neq= 0 \\
\Longleftrightarrow & H_0 : \mathbf{C}\boldsymbol{\beta} = \mathbf{0}_5 \quad \text{versus} \quad H_1 : \mathbf{C}\boldsymbol{\beta} \neq \mathbf{0}_5
\end{aligned}
$$

It is clear that

$$
\mathbf{C}\boldsymbol{\beta} = \mathbf{0}_5 \Longleftrightarrow \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \end{bmatrix} = \mathbf{0}_5
$$

The corresponding test statistic is

$$
F = \frac{(\mathbf{C}\widehat{\boldsymbol{\beta}})^\top \left[\mathbf{C}(\mathbb{X}^\top\mathbb{X})^{-1}\mathbf{C}^\top\right]^{-1}(\mathbf{C}\widehat{\boldsymbol{\beta}})/q}{\widehat{\sigma}^2} \sim F_{q,n-r}, \qquad q = 5.
$$

Solutions

Method 1: Nested model comparison

- Fit the reduced model (intercept only)

```
null_model <- lm(attack ~ 1, data = pok)
```

- Perform the overall test with `anova()`

```
global_test <- anova(null_model, full_model)
global_test
```

```
Analysis of Variance Table

Model 1: attack ~ 1
Model 2: attack ~ speed + defense + hp + sp_attack + sp_def
  Res.Df    RSS Df Sum of Sq   F              Pr(>F)
1    799 841731
2    794 471595  5    370136 125 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
global_test$F
```

```
[1]     NA 124.64
```

```
summary(full_model)[["fstatistic"]]
```

```
 value  numdf  dendf
124.64   5.00 794.00
```

Method 2: General linear hypothesis test

- Compute F statistic with `linearHypothesis()` from `{car}`

$$F = \frac{(\mathbf{C}\hat{\boldsymbol{\beta}})^\top [\mathbf{C}(\mathbb{X}^\top \mathbb{X})^{-1}\mathbf{C}^\top]^{-1}(\mathbf{C}\hat{\boldsymbol{\beta}})/q}{\hat{\sigma}^2}$$

```r
predictors
```

```
[1] "speed"     "defense"   "hp"        "sp_attack" "sp_def"
```

```r
linearHypothesis(full_model, predictors)
```

```
Linear hypothesis test:
speed = 0
defense = 0
hp = 0
sp_attack = 0
sp_def = 0

Model 1: restricted model
Model 2: attack ~ speed + defense + hp + sp_attack + sp_def

  Res.Df    RSS Df Sum of Sq   F               Pr(>F)
1    799 841731
2    794 471595  5    370136 125 <0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- We can also compute F statistic manually

$$F = \frac{(\mathbf{C}\hat{\beta})^\top [\mathbf{C}(\mathbb{X}^\top \mathbb{X})^{-1} \mathbf{C}^\top]^{-1} (\mathbf{C}\hat{\beta})/q}{\hat{\sigma}^2}$$

```r
C <- cbind(rep(0, 5), diag(nrow = 5))
betas <- coef(full_model)

# Design matrix
X <- model.matrix(full_model)

# Residual variance estimate (σ^2-hat)
sigma2_hat <- sigma(full_model)^2

# Number of restrictions
q <- nrow(C)

# Numerator: (C beta_hat)
C_beta <- C %*% betas

# Middle matrix: [C (X'X)^(-1) C']^(-1)
middle <- solve(C %*% solve(t(X) %*% X) %*% t(C))

# F-statistic
F_stat <- as.numeric(t(C_beta) %*% middle %*% C_beta / (q * sigma2_hat))
```

- F-statistic

```r
F_stat
```

```
[1] 124.64
```

- Critical value and p-value

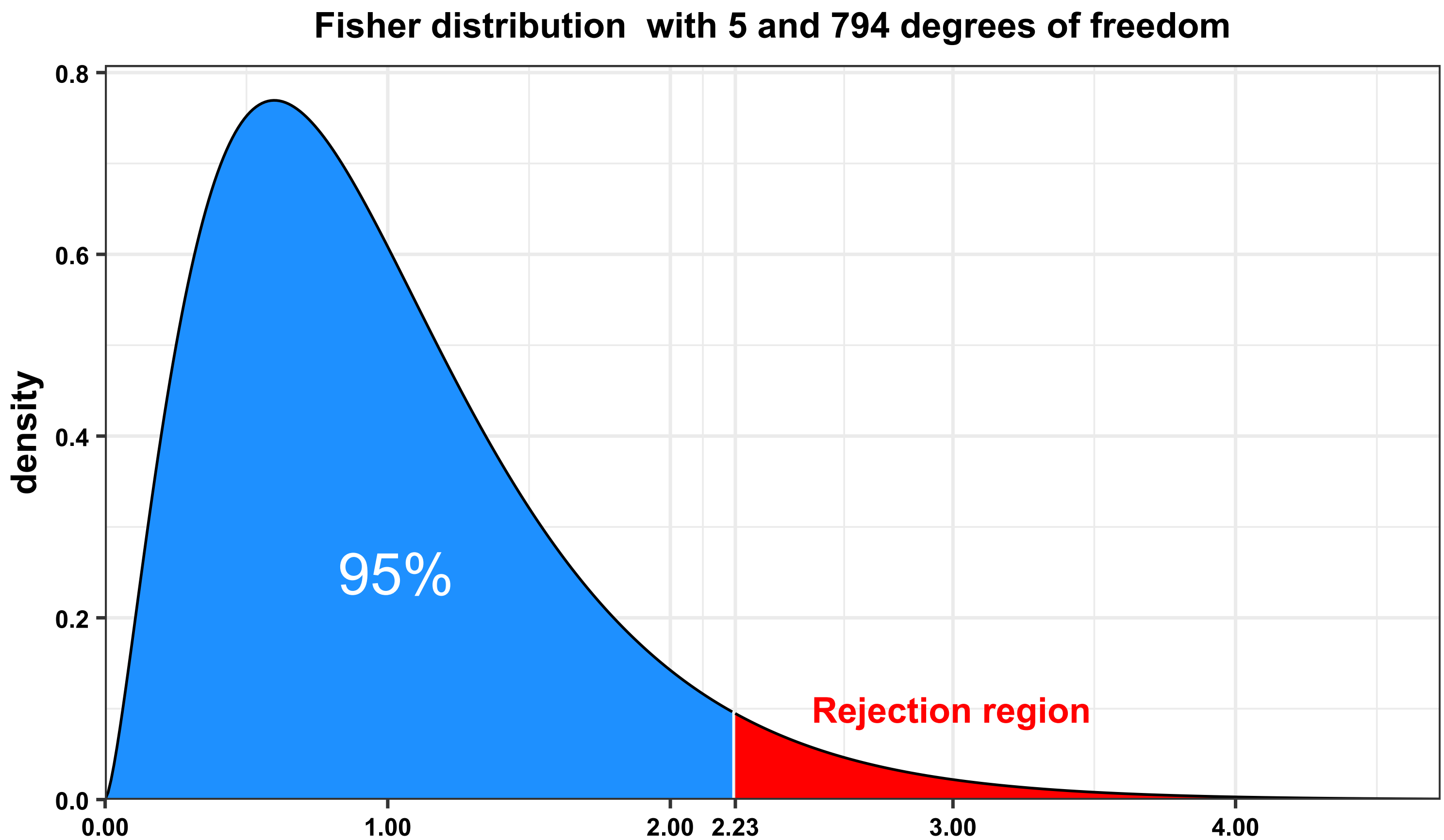
```r
# 95% quantile of F(q, n-r)
qf(0.95, q, df.residual(full_model), lower.tail = TRUE)
```

```
[1] 2.2254
```

```r
# p-value
pf(F_stat, q, df.residual(full_model), lower.tail = FALSE) |> label_scientific()()
```

```
[1] "2.27e-97"
```

- Visualize rejection region of the Fisher distribution



Question 7. Indices of model performance for regression

Compute indices of performance for the `full_model`. **Hint:** `glance()`, `model_performance()`

Solutions

- With `glance()` from `{broom}`

```
glance(full_model)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance df.residual
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>    <dbl>       <int>
1     0.440         0.436  24.4      125. 2.27e-97     5 -3687. 7388. 7421.  471595.         794
```

$$R^2 = \frac{\text{MSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}} \qquad R_a^2 = 1 - \frac{(n-1)\text{RSS}}{(n-m)\text{TSS}} \quad (m = p+1)$$

$$\text{AIC}(m) = n \log\left(\frac{\text{RSS}(m)}{n}\right) + 2m \qquad \text{BIC}(m) = n \log\left(\frac{\text{RSS}(m)}{n}\right) + \log(n) \times m$$

$$\text{deviance}(m) = \text{RSS}(m) \quad \text{df.residual}(m) = n - m \quad \text{sigma}(m) = \sqrt{\frac{\text{RSS}}{n-m}} = \hat{\sigma}$$

- With `model_performance()` from `{performance}`

```
model_performance(full_model)
```

```
# Indices of model performance

AIC    |   AICc |    BIC |    R2 | R2 (adj.) |   RMSE |  Sigma
---------------------------------------------------------------
7387.7 | 7387.9 | 7420.5 | 0.440 |     0.436 | 24.279 | 24.371
```

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- **Other useful functions**

```r
AIC(full_model)
```

```
[1] 7387.7
```

```r
BIC(full_model)
```

```
[1] 7420.5
```

```r
r2(full_model, ci = 0.95) # {performance}
```

```
        R2: 0.440 [0.384, 0.487]
   adj. R2: 0.436 [0.380, 0.484]
```

```r
summary(full_model)[["r.squared"]]
```

```
[1] 0.43973
```

```r
summary(full_model)[["adj.r.squared"]]
```

```
[1] 0.4362
```

Question 8. Joint hypothesis test

Consider the full regression model `full_model`

$$\texttt{attack} = \beta_0 + \beta_1\texttt{speed} + \beta_2\texttt{defense} + \beta_3\texttt{hp} + \beta_4\texttt{sp\_attack} + \beta_5\texttt{sp\_def} + \varepsilon$$

We want to test jointly whether the coefficients on `sp_attack` and `sp_def` are equal to zero:

$$H_0 : \beta_4 = \beta_5 = 0 \quad \text{versus} \quad H_1 : \text{at least one of } \beta_4, \beta_5 \text{ is nonzero.}$$

Hints:

- Compare the **full model** with a **restricted model** (without `sp_attack` and `sp_def`) using an F-test (`anova()`).
- Use a **joint Wald test** (`linearHypothesis()`, `waldtest()`).

Solutions

Method 1: Nested model comparison

- Fit the restricted model, `res_model`

```
res_model <- lm(attack ~ speed + defense + hp, data = pok)
```

- Compare with the full model using `anova()`:

```
anova(res_model, full_model) |> qTBL()
```

```
# A tibble: 2 x 6
  Res.Df     RSS    Df `Sum of Sq`     F  `Pr(>F)`
   <dbl>   <dbl> <dbl>       <dbl> <dbl>     <dbl>
1    796 502738.    NA         NA    NA   NA
2    794 471595.     2      31143.  26.2  9.42e-12
```

- Compute F statistics by hand: $F = \frac{[\text{RSS}(m_0) - \text{RSS}(m_1)]/(p-q)}{\text{RSS}(m_1)/(n-r)} \sim F_{p-q,n-r} \quad (\text{under } H_0)$

```
rss0 <- deviance(res_model)
df0 <- df.residual(res_model)
rss1 <- deviance(full_model)
df1 <- df.residual(full_model)
fstat <- ((rss0 - rss1) / (df0 - df1)) / (rss1 / df1)
fstat
```

```
[1] 26.217
```

- Compute p-value

```r
pf(fstat, df0 - df1, df1, lower.tail = FALSE)
```

```
[1] 0.0000000000094234
```

Method 2: Wald-type tests

- Use `linearHypothesis()` from `{car}`

```r
linearHypothesis(full_model, c("sp_attack = 0", "sp_def = 0"))
```

```
Linear hypothesis test:
sp_attack = 0
sp_def = 0

Model 1: restricted model
Model 2: attack ~ speed + defense + hp + sp_attack + sp_def

  Res.Df    RSS Df Sum of Sq    F          Pr(>F)
1    796 502738
2    794 471595  2     31143 26.2 0.0000000000094 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Or use `waldtest()` from `{lmtest}`

```r
# Compare restricted vs full model
waldtest(res_model, full_model, test = "F")


# Test restrictions directly
waldtest(full_model, c("sp_attack", "sp_def"), test = "F")
```

```
Wald test

Model 1: attack ~ speed + defense + hp + sp_attack + sp_def
Model 2: attack ~ speed + defense + hp
  Res.Df Df    F          Pr(>F)
1    794
2    796 -2 26.2 0.0000000000094 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Question 9. Prediction & Intervals

Consider `full_model`

1. Compute 95% Confidence Interval for the mean $\mathbb{E}(\texttt{attack})$ given $\texttt{speed} = \texttt{30}, \texttt{70}, \texttt{110}, \texttt{150}$ and fixing the other predictors at their mean

Hint: `predict(..., interval = "confidence")`, `estimate_expectation()`

2. Suppose a new Pokémon is created with the following characteristics :

```
    speed   defense        hp sp_attack    sp_def
       50        42       100       135        60
```

Predict the `attack` for Pokémon and the appropriate 95%CI.

Solutions

- **Create grid of values**: $\texttt{speed} = \texttt{30}, \texttt{70}, \texttt{110}, \texttt{150}$ and fixing the other predictors at their mean with `get_datagrid()` from **{insight}**

```
grid1 <- select(pok, all_of(predictors)) |>
  get_datagrid(by = "speed = seq(30, 150, 40)", numerics = "integer")

grid1
```

```
Visualisation Grid

speed | defense | hp | sp_attack | sp_def
-----------------------------------------
30    |      74 | 69 |        73 |     72
70    |      74 | 69 |        73 |     72
110   |      74 | 69 |        73 |     72
150   |      74 | 69 |        73 |     72

Maintained constant: defense, hp, sp_attack, sp_def
```

- Predicted and 95% Confidence Interval for the prediction

```r
estimate_expectation(full_model, data = grid1, ci = 0.95)
```

```
Model-based Predictions

speed | Predicted |   SE |           95% CI
--------------------------------------------
30    |     65.88 | 1.56 | [ 62.82,  68.95]
70    |     79.57 | 0.86 | [ 77.88,  81.27]
110   |     93.26 | 1.66 | [ 90.00,  96.51]
150   |    106.95 | 2.91 | [101.24, 112.65]

Variable predicted: attack
Predictors modulated: speed = seq(30, 150, 40)
Predictors controlled: defense (74), hp (69), sp_attack (73), sp_def (72)
```

- We now predict the `attack` of a Pokèmon with the following characteristics

```
    speed   defense        hp sp_attack     sp_def
       50        42       100       135         60
```

```r
grid2 <- c(speed = 50, defense = 42, hp = 100, sp_attack = 135, sp_def = 60) |>
  as_tibble_row()

grid2
```

```
# A tibble: 1 x 5
  speed defense    hp sp_attack sp_def
  <dbl>   <dbl> <dbl>     <dbl>  <dbl>
1    50      42   100       135     60
```

- Prediction interval with `estimate_prediction()`

```r
estimate_prediction(full_model, data = grid2, ci = 0.95)
```

```
Model-based Predictions

speed | defense |  hp | sp_attack | sp_def | Predicted |    SE |          95% CI
--------------------------------------------------------------------------------
50    |      42 | 100 |       135 |     60 |     82.84 | 24.56 | [34.63, 131.04]

Variable predicted: attack
```

Question 10. Residual diagnostics

Note: Standardized vs Studentized residuals

- Let denote by h_{ij} the element of the projector $P_{\mathbb{X}} = H_{\mathbb{X}}$ such that $P_{\mathbb{X}} = H_{\mathbb{X}} = [h_{ij}]$
- The diagonal elements $h_{ii} \in [0, 1]$ are called the *leverages*
- If $h_{ii} > 2p/n$ (sometimes $h_{ii} > 3p/n$), then the observation i is consider an *outlier*

- **Standardized residuals** (from `rstandard()`)
  Raw residuals are rescaled by their estimated standard deviation, taking into account leverage.

$$\hat{r}_i = \frac{\hat{\varepsilon}_i}{\hat{\sigma}\sqrt{1 - h_{ii}}}$$

where $\hat{\varepsilon}_i$ is the raw residual and h_{ii} is the leverage of observation i.
These make residuals roughly comparable across observations.

- **Studentized residuals** (from `rstudent()`)
  Go one step further: each residual is scaled using a variance estimate that **excludes the** i-th observation. This gives more accurate standard errors and makes large outliers easier to detect.

$$t_i^* = \frac{\hat{\varepsilon}_i}{\hat{\sigma}_{(-i)}\sqrt{1 - h_{ii}}}$$

where $\hat{\sigma}_{(-i)}$ is the error standard deviation estimated without observation i.

Using `full_model` and functions from the file `helper_functions.R`:

1. Plot residuals vs fitted values and vs each predictor speed, defense, hp, sp_attack, sp_def.
2. Plot $\sqrt{|\text{Standardized residuals}|}$ vs fitted values and vs each predictor.
3. Plot studentized residuals vs fitted values and vs each predictor.
4. Plot residuals in the order of observation (to detect dependence).
5. Plot a histogram of the standardized residuals.
6. Perform a normality test on standardized residuals.
7. Plot a normal Q-Q plot of standardized residuals.
8. Perform the Breusch–Pagan test for heteroskedasticity.
9. Perform the Durbin–Watson test on the residuals.

Solutions

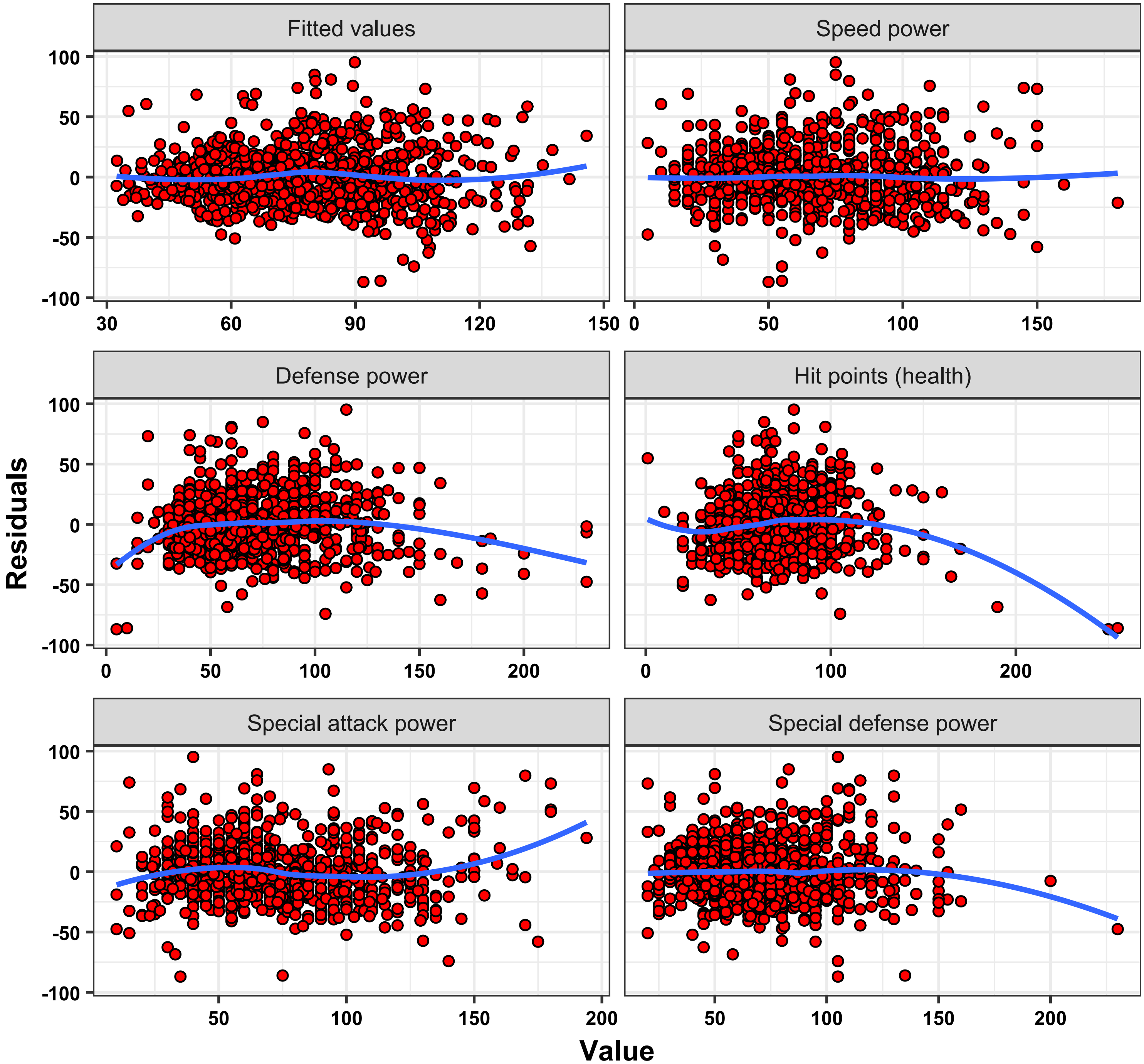
- **Prepare residual diagnostics dataset**

```r
augment_full <- augment(full_model)
head(augment_full)
```

```
# A tibble: 6 x 12
  attack speed defense    hp sp_attack sp_def .fitted .resid    .hat .sigma    .cooksd .std.resid
   <dbl> <dbl>   <dbl> <dbl>     <dbl>  <dbl>   <dbl>  <dbl>   <dbl>  <dbl>      <dbl>      <dbl>
1     49    45      49    45        65     65    51.1  -2.08 0.00404   24.4 0.00000492    -0.0854
2     62    60      63    60        80     80    66.6  -4.60 0.00244   24.4 0.0000145     -0.189
3     82    80      83    80       100    100    87.9  -5.92 0.00277   24.4 0.0000274     -0.243
4    100    80     123    80       122    120   105.   -4.67 0.00691   24.4 0.0000430     -0.192
5     52    65      43    39        60     50    56.3  -4.26 0.00371   24.4 0.0000190     -0.175
6     64    80      58    58        80     65    74.6 -10.6  0.00211   24.4 0.0000663     -0.433
```

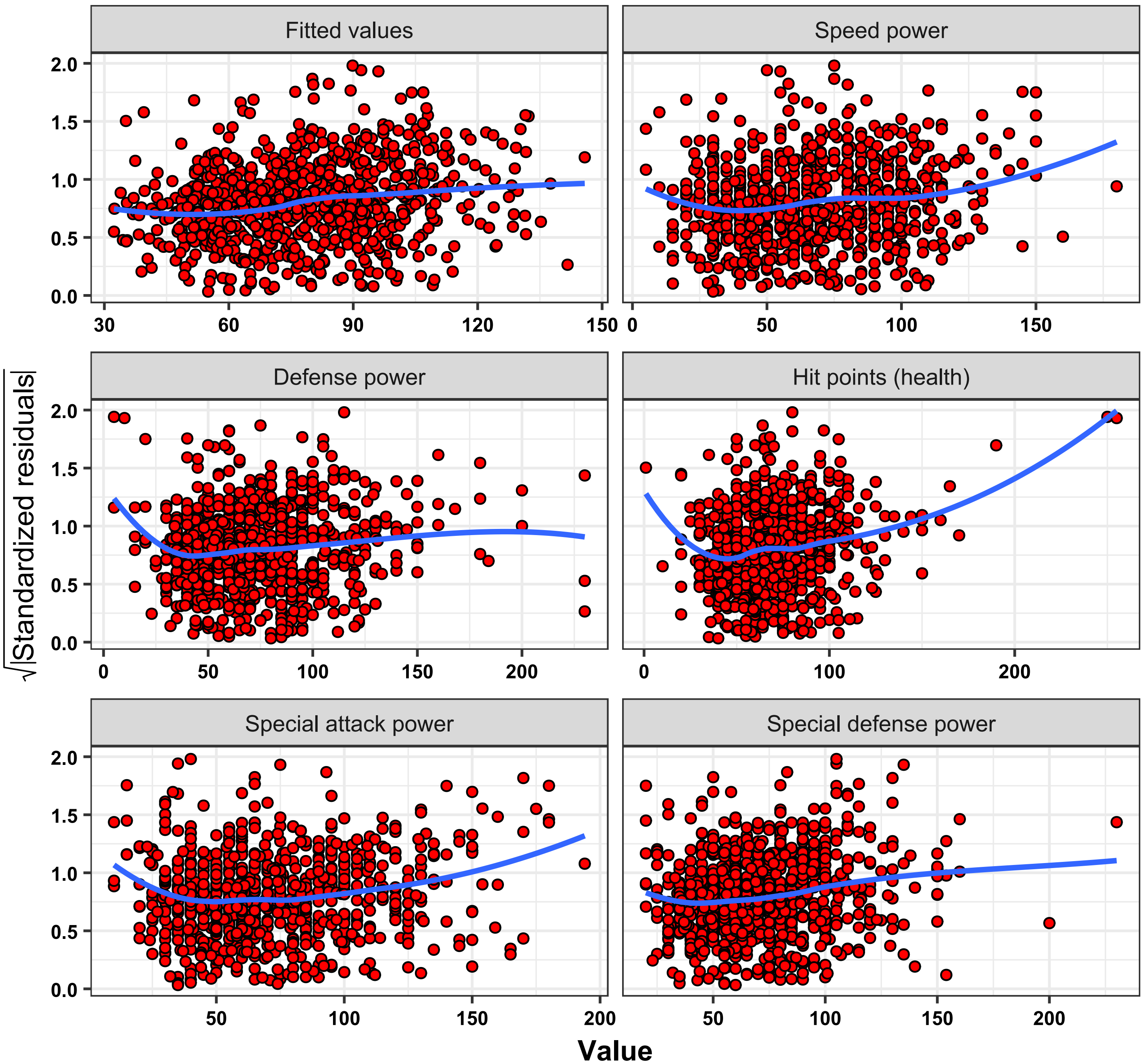
- **Residuals vs fitted values and predictors**

```
resid_vs_predictors(model = full_model, predictors = predictors)
```



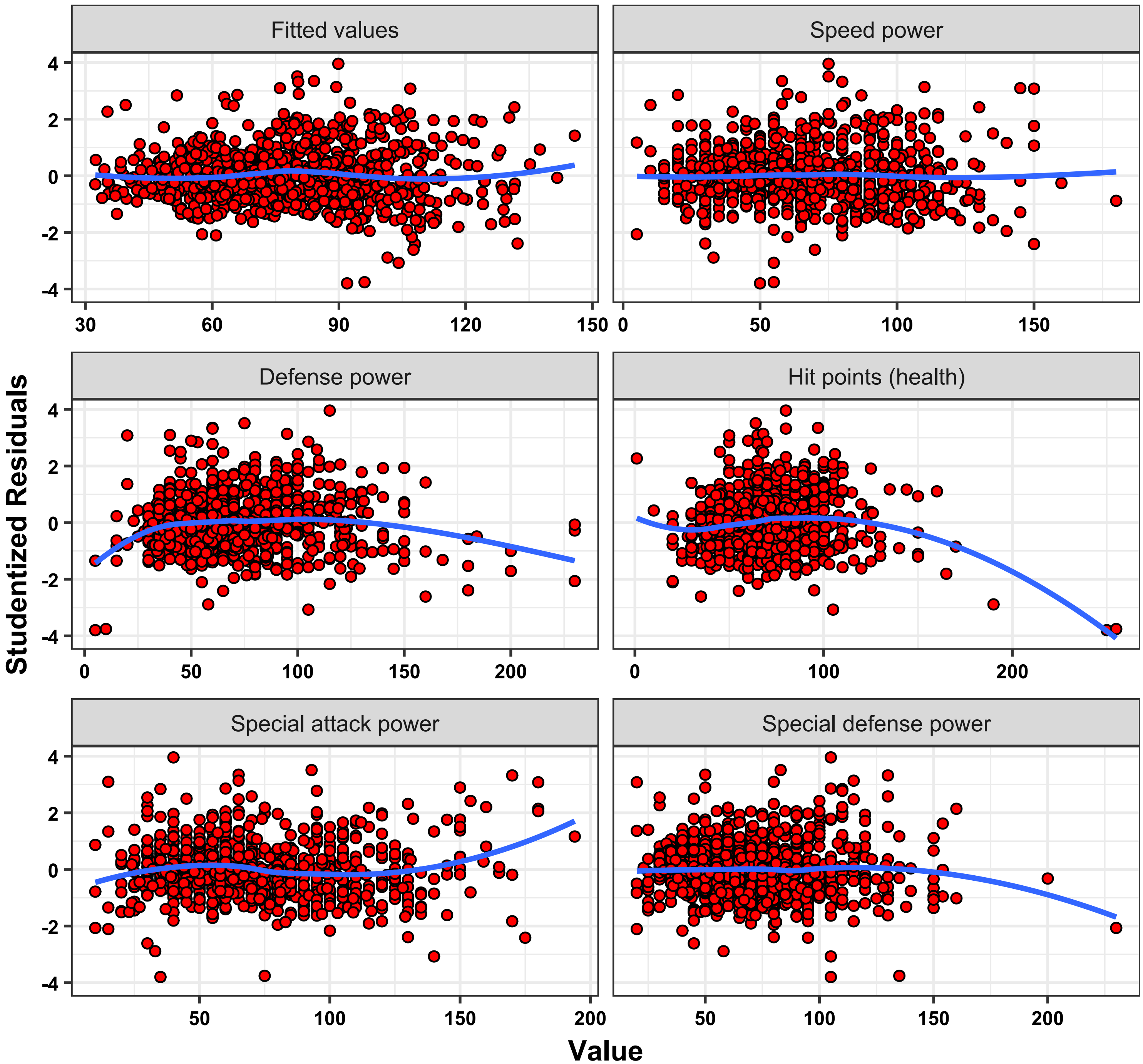
- $\sqrt{|\text{Standardized residuals}|}$ vs fitted values and predictors

```
resid_stand_vs_predictors(model = full_model, predictors = predictors)
```



- **Studentized residuals vs fitted values and predictors**

```
resid_stud_vs_predictors(full_model, predictors)
```



- **Residuals vs observation order**

```r
p1 <- resid_vs_order(full_model)
```

- **Histogram of standardized residuals**

```r
p2 <- resid_stand_hist(full_model)
```

- **Density of standardized residuals**

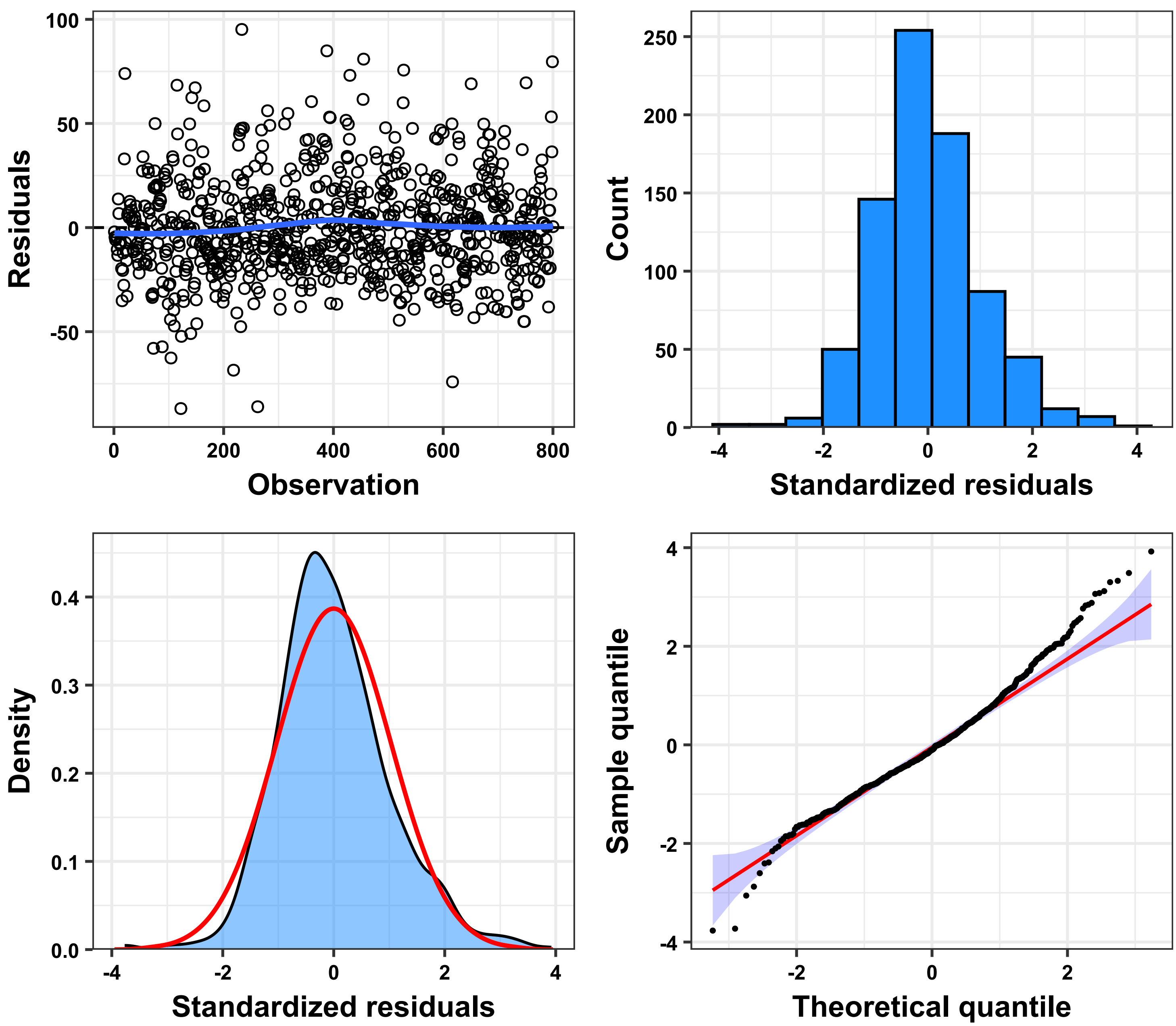
```r
p3 <- resid_stand_dens(full_model)
```

- **Normal Q–Q plot**

```r
p4 <- resid_stand_qq(full_model)
```

- We combine the 4 plots with the help of `{patchwork}`

```
p1 + p2 + p3 + p4
```



- **Breusch–Pagan test for heteroskedasticity**

```r
ncvTest(full_model)
```

```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 37.223, Df = 1, p = 0.00000000105
```

The **Breusch–Pagan test** strongly rejects the null hypothesis ($p < 10^{-6}$), but with a large sample, even small deviations from perfect homoscedasticity will appear "significant."

From the residual and scale–location plots, variance seems roughly constant with only mild increases for certain predictors (e.g., `hp`, `sp_def`).

Conclusion:

The test result likely overstates the problem. There is *mild heteroscedasticity*, but not enough to invalidate the model.

- **Durbin–Watson test**

```r
durbinWatsonTest(full_model)
```

```
 lag Autocorrelation D-W Statistic p-value
   1         0.30572        1.3886       0
 Alternative hypothesis: rho != 0
```

The **Durbin–Watson test** gave a statistic of about 1.39 with a very small p-value, suggesting positive autocorrelation.

However, with ($n \approx 800$), the test becomes *too powerful* and flags even trivial correlations as significant.

Moreover, this dataset is not a time series (observations are not ordered chronologically), so the detected autocorrelation may reflect mild structural grouping rather than temporal dependence.

Conclusion:

Although the test is significant, there is no visible pattern in the residual plots.
Residuals appear *approximately independent*, and the practical impact on OLS validity is minimal.

- **Shapiro–Wilk test of residual normality**

```
augment(full_model) |>
  shapiro_test(.std.resid)
```

```
# A tibble: 1 x 3
  variable   statistic            p
  <chr>          <dbl>        <dbl>
1 .std.resid     0.981 0.0000000102
```

The **Shapiro–Wilk test** produced a tiny p-value ($p < 10^{-7}$), but with large n, it detects even negligible deviations.

The histogram and density of standardized residuals appear symmetric and bell-shaped, and the Q–Q plot shows only mild tail deviations.

Conclusion:

Although the Shapiro–Wilk test rejects normality, this is expected with large samples.
The residual distribution is *approximately normal*, with symmetry and unimodality clearly visible.
OLS estimates remain unbiased and efficient, and inference remains valid due to the Central Limit Theorem.

Overall Assessment

Assumption	Test Result	Visual Assessment	Practical Verdict
Independence	DW significant	No clear pattern	Acceptable
Homoscedasticity	BP significant	Mild variance changes	Acceptable
Normality	SW significant	Roughly symmetric	Acceptable

Session Info

Package	Version
broom	1.0.10
car	3.1-3
collapse	2.1.4
correlation	0.8.8
datawizard	1.3.0
effectsize	1.0.1
GGally	2.4.0
ggfortify	0.4.19
ggpubr	0.6.2
glue	1.8.0
gtsummary	2.4.0
insight	1.4.2
kableExtra	1.4.0
lmtest	0.9-40
matrixTests	0.2.3.1
modelbased	0.13.0
parameters	0.28.2
patchwork	1.3.2
performance	0.15.2
qqplotr	0.0.7
rstatix	0.7.3
scales	1.4.0
see	0.12.0
tidyverse	2.0.0