R Recitation - 6 January: Correlation (Parametric & Non-Parametric) + Linear Regression (Full Walkthrough)

Burcu

06 January 2026

Contents

Learning goals

By the end of this recitation, you should be able to:

1. Compute and interpret correlation using:

    - Pearson (parametric)
    - Spearman and Kendall (non-parametric)

2. Build a linear regression model step-by-step in R:
    - Specify the model
    - Inspect coefficients and model fit
    - Compare nested models
3. Diagnose regression assumptions and common problems:
    - Linearity
    - Independence (conceptual + when it matters)
    - Homoscedasticity
    - Normality of residuals
    - Outliers and influential observations
    - Multicollinearity (for multiple regression)
4. Report results clearly using standard statistics language.

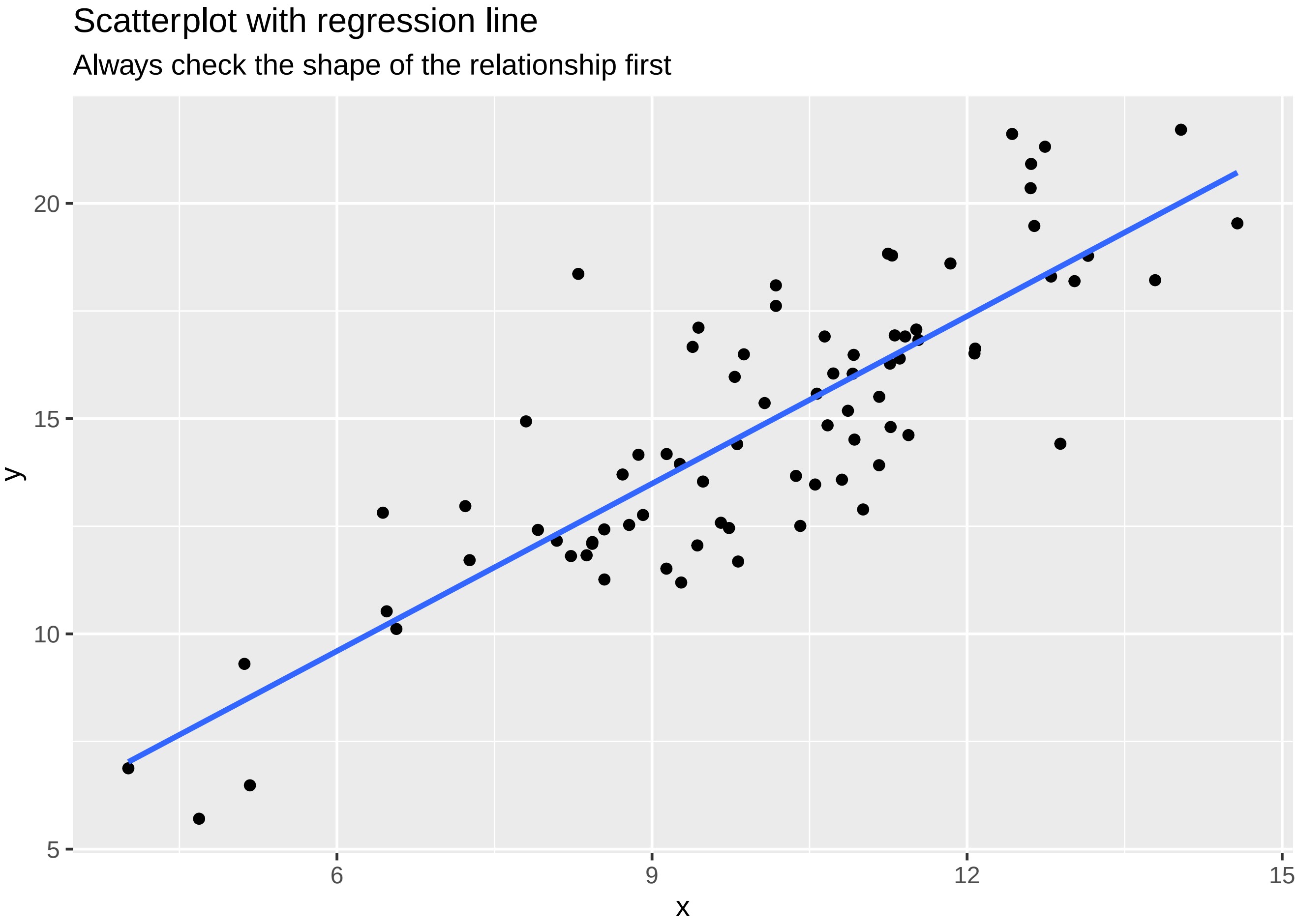
Correlation (briefly)

Why start with a plot?

Correlation is a number, but *relationships are visual.* Always start with a scatterplot.

We'll work with a small simulated example to illustrate all methods cleanly.

```r
set.seed(42)
n <- 80

# Create an x variable

x <- rnorm(n, mean = 10, sd = 2)

# Create a y variable with a roughly linear relationship + noise

y <- 3 + 1.2 * x + rnorm(n, mean = 0, sd = 2)

df_corr <- tibble(x = x, y = y)

ggplot(df_corr, aes(x, y)) +
geom_point() +
geom_smooth(method = "lm", se = FALSE) +
labs(title = "Scatterplot with regression line",
subtitle = "Always check the shape of the relationship first")
```



Pearson correlation (parametric)

Pearson correlation measures *linear association* between two continuous variables.

Typical conditions (practical framing)

- The relationship is approximately linear.
- Extreme outliers can distort Pearson strongly.
- Normality is not a strict requirement for using Pearson in all cases, but it matters more for small samples and for inference.

```
cor(df_corr$x, df_corr$y, method = "pearson")
```

```
## [1] 0.8364027
```

```
cor.test(df_corr$x, df_corr$y, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_corr$x and df_corr$y
## t = 13.477, df = 78, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7555312 0.8921649
## sample estimates:
##       cor
## 0.8364027
```

How to report (template)

"There was a (positive/negative) linear association between X and Y, Pearson's r(df) = …, p = …, 95% CI […, …]."

Spearman and Kendall (non-parametric)

Non-parametric correlation is useful when:

- The relationship is monotonic but not linear,
- The data are ordinal, or
- You want a rank-based measure less sensitive to outliers / non-normality.

Spearman (rank correlation; monotonic relationships)

```r
cor(df_corr$x, df_corr$y, method = "spearman")
```

```
## [1] 0.7943741
```

```r
cor.test(df_corr$x, df_corr$y, method = "spearman", exact = FALSE)
```

```
##
##  Spearman's rank correlation rho
##
## data:  df_corr$x and df_corr$y
## S = 17544, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.7943741
```

Kendall (based on concordant/discordant pairs; robust in small n / ties)

```r
cor(df_corr$x, df_corr$y, method = "kendall")
```

```
## [1] 0.6101266
```

```r
cor.test(df_corr$x, df_corr$y, method = "kendall", exact = FALSE)
```

```
##
##  Kendall's rank correlation tau
##
```

```
## data:  df_corr$x and df_corr$y
## z = 8.0102, p-value = 1.145e-15
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##       tau
## 0.6101266
```

Quick comparison: when to use which?

- **Pearson**: linear association, continuous variables, no severe outliers, interpretability in linear units is aligned with regression.
- **Spearman**: monotonic association (not necessarily linear), ordinal data, more robust to non-normality and some outliers.
- **Kendall**: similar goals as Spearman; can be preferable with small samples and many ties.

Chi-square Test

What question does the chi-square test answer?

The chi-square test of independence examines whether two categorical variables are statistically associated.

Conceptually, this is the categorical analogue of correlation: - Correlation → association between continuous variables - Chi-square → association between categorical variables

```r
dat <- as.data.frame(Titanic)

# Class × Survival
tbl <- xtabs(Freq ~ Class + Survived, data = dat)
tbl
```

```
##       Survived
## Class   No Yes
##   1st  122 203
##   2nd  167 118
##   3rd  528 178
##   Crew 673 212
```

```r
chisq.test(tbl)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tbl
## X-squared = 190.4, df = 3, p-value < 2.2e-16
```

```r
chisq.test(tbl)$expected
```

```
##       Survived
## Class        No       Yes
##   1st  220.0136 104.98637
##   2nd  192.9350  92.06497
##   3rd  477.9373 228.06270
##   Crew 599.1140 285.88596
```

Assumptions of the chi-square test

- Observations are independent
- Expected cell counts should generally be 5

When expected counts are small, the chi-square approximation may be inaccurate.

Fisher's Exact Test (small samples)

When expected cell counts are small, Fisher's Exact Test is preferred.

```r
fisher.test(tbl, simulate.p.value = TRUE, B = 20000)
```

```
##
##  Fisher's Exact Test for Count Data with simulated p-value (based on
##  20000 replicates)
##
## data:  tbl
## p-value = 5e-05
## alternative hypothesis: two.sided
```

Linear regression (walkthrough)

Regression is not just "a line": it is a *model* that explains/predicts an outcome using one or more predictors, and it comes with assumptions we must check.

We will use a real dataset (mtcars) for a concrete end-to-end demonstration.

Data and question

We will model fuel efficiency (mpg) using:

- wt (car weight)

- hp (horsepower)

Question: How do weight and horsepower relate to miles per gallon?

```r
df <- mtcars %>%
as_tibble(rownames = "car") %>%
select(car, mpg, wt, hp, cyl)

glimpse(df)
```

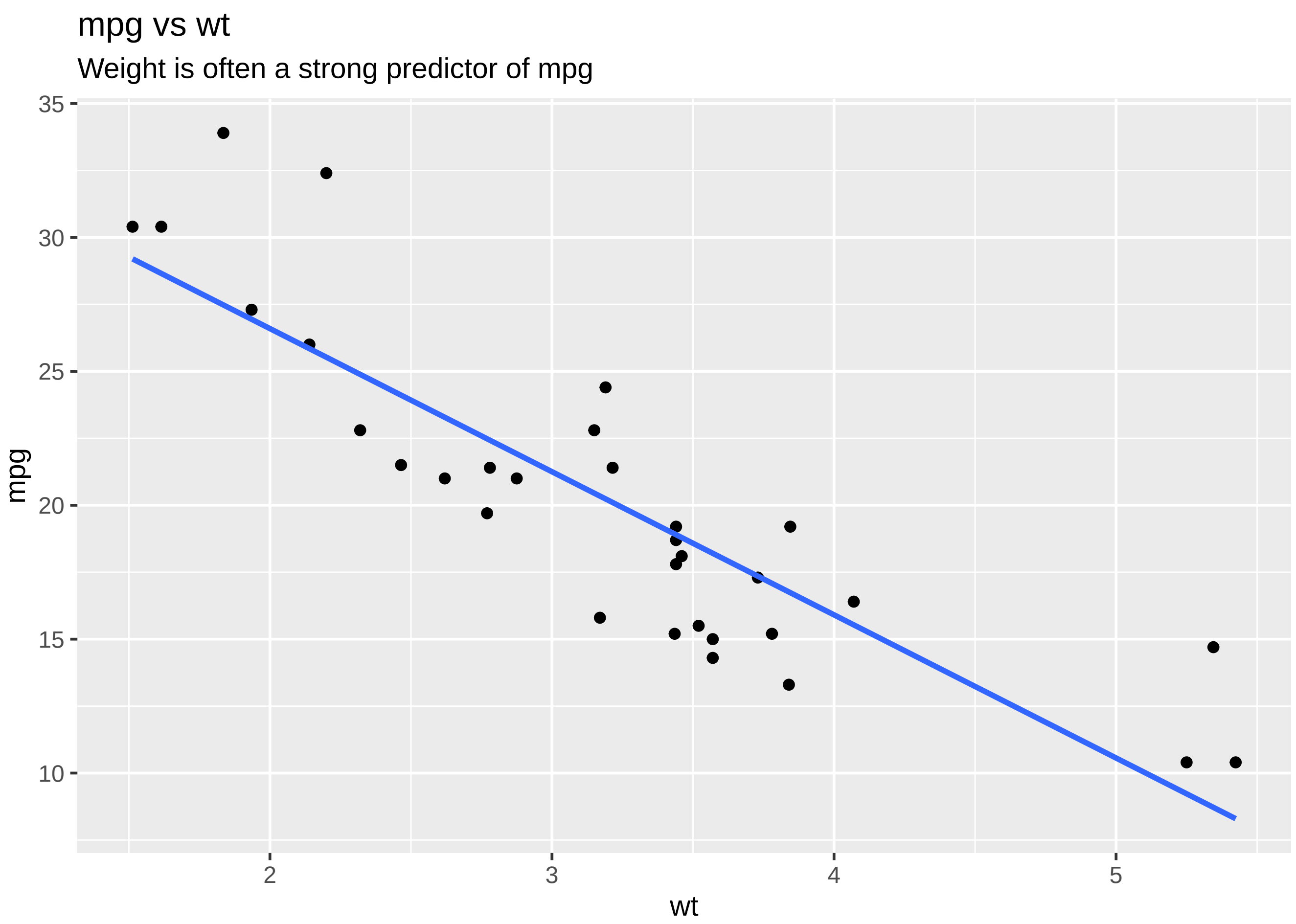
```
## Rows: 32
## Columns: 5
## $ car <chr> "Mazda RX4", "Mazda RX4 Wag", "Datsun 710", "Hornet 4 Drive", "Hor~
## $ mpg <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, ~
## $ wt  <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3.150, 3.4~
## $ hp  <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, 180, 180,~
## $ cyl <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 8, 4, 4, 4, 4, 8, ~
```

```r
summary(df)
```

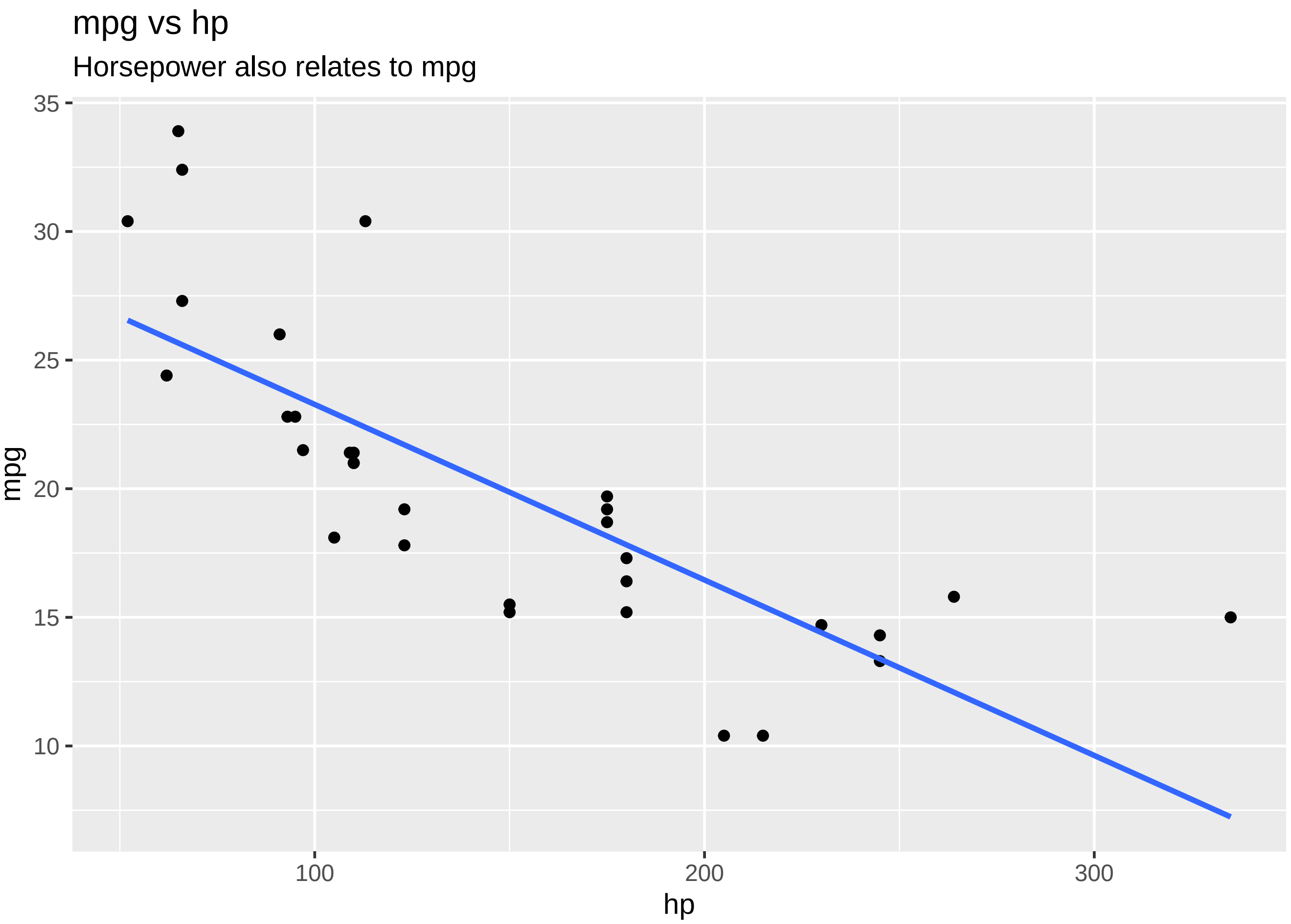
```
##      car                 mpg              wt              hp       
##  Length:32          Min.   :10.40   Min.   :1.513   Min.   : 52.0  
##  Class :character   1st Qu.:15.43   1st Qu.:2.581   1st Qu.: 96.5  
##  Mode  :character   Median :19.20   Median :3.325   Median :123.0  
##                     Mean   :20.09   Mean   :3.217   Mean   :146.7  
##                     3rd Qu.:22.80   3rd Qu.:3.610   3rd Qu.:180.0  
##                     Max.   :33.90   Max.   :5.424   Max.   :335.0  
##       cyl       
##  Min.   :4.000  
##  1st Qu.:4.000  
##  Median :6.000  
##  Mean   :6.188  
##  3rd Qu.:8.000  
##  Max.   :8.000
```

First: visualize relationships

```r
ggplot(df, aes(wt, mpg)) +
geom_point() +
geom_smooth(method = "lm", se = FALSE) +
labs(title = "mpg vs wt", subtitle = "Weight is often a strong predictor of mpg")
```



```
ggplot(df, aes(hp, mpg)) +
geom_point() +
geom_smooth(method = "lm", se = FALSE) +
labs(title = "mpg vs hp", subtitle = "Horsepower also relates to mpg")
```



The simple linear regression model (one predictor)

Model specification

A simple linear regression with one predictor is:

$mpg_i = b_0 + b_1 \cdot wt_i + \epsilon_i$

where:

- b_0 is the intercept,
- b_1 is the slope for wt
- ϵ_i are residual errors.

Fit the model in R

```
m1 <- lm(mpg ~ wt, data = df)
summary(m1)
```

```
##
## Call:
## lm(formula = mpg ~ wt, data = df)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5432 -2.3647 -0.1252  1.4096  6.8727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
## wt           -5.3445     0.5591  -9.559 1.29e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.046 on 30 degrees of freedom
## Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
## F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

Interpret coefficients

- **Intercept (b_0):** predicted mpg when $wt = 0$ (often not meaningful if wt=0 is outside the data range; but it is part of the line).
- **Slope (b_1):** expected change in mpg for a 1-unit increase in wt (here, wt is 1000 lbs in `mtcars` units).

Get a clean coefficient table

```
tidy(m1, conf.int = TRUE)
```

```
## # A tibble: 2 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)    37.3      1.88      19.9  8.24e-19    33.5      41.1
## 2 wt             -5.34     0.559     -9.56 1.29e-10    -6.49     -4.20
```

Model fit: R-squared and residual standard error

The `summary(m1)` output includes:

- **Multiple R-squared:** proportion of variance in mpg explained by wt.
- **Adjusted R-squared:** penalizes for extra predictors (important later).
- **Residual standard error (RSE):** typical size of prediction errors (in mpg).

Extract key fit metrics programmatically:

```
glance(m1) %>%
select(r.squared, adj.r.squared, sigma, statistic, p.value, df.residual)
```

```
## # A tibble: 1 x 6
##   r.squared adj.r.squared sigma statistic  p.value df.residual
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl>       <int>
## 1     0.753         0.745  3.05      91.4 1.29e-10          30
```

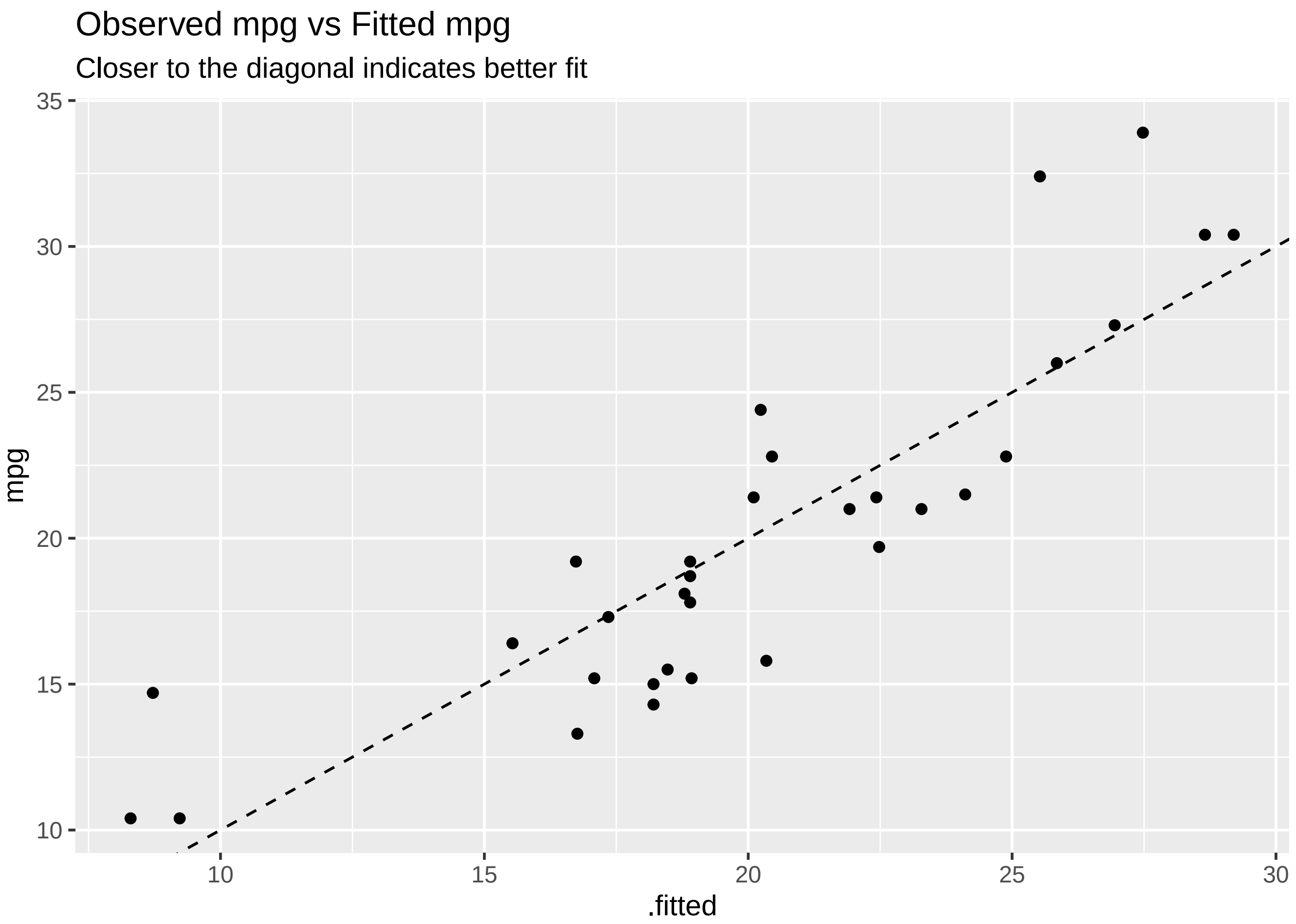
Predictions (fitted values) and residuals

```r
df_aug <- augment(m1)  # adds .fitted and .resid columns
head(df_aug)
```

```
## # A tibble: 6 x 8
##     mpg    wt .fitted .resid   .hat .sigma   .cooksd .std.resid
##   <dbl> <dbl>   <dbl>  <dbl>  <dbl>  <dbl>     <dbl>      <dbl>
## 1  21    2.62    23.3 -2.28  0.0433   3.07 0.0133       -0.766
## 2  21    2.88    21.9 -0.920 0.0352   3.09 0.00172      -0.307
## 3  22.8  2.32    24.9 -2.09  0.0584   3.07 0.0154       -0.706
## 4  21.4  3.22    20.1  1.30  0.0313   3.09 0.00302       0.433
## 5  18.7  3.44    18.9 -0.200 0.0329   3.10 0.0000760    -0.0668
## 6  18.1  3.46    18.8 -0.693 0.0332   3.10 0.000921     -0.231
```

Plot fitted vs observed:

```r
ggplot(df_aug, aes(.fitted, mpg)) +
geom_point() +
geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
labs(title = "Observed mpg vs Fitted mpg",
subtitle = "Closer to the diagonal indicates better fit")
```



Multiple regression (step-by-step model building)

Now we add horsepower:

$mpg_i = b_0 + b_1 \cdot wt_i + b_2 \cdot hp_i + \epsilon_i$

```r
m2 <- lm(mpg ~ wt + hp, data = df)
summary(m2)
```

```
##
## Call:
## lm(formula = mpg ~ wt + hp, data = df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.941 -1.600 -0.182  1.050  5.854
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
## wt          -3.87783    0.63273  -6.129 1.12e-06 ***
## hp          -0.03177    0.00903  -3.519  0.00145 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.593 on 29 degrees of freedom
## Multiple R-squared:  0.8268, Adjusted R-squared:  0.8148
## F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12
```

```r
tidy(m2, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)  37.2      1.60        23.3  2.57e-20  34.0      40.5
## 2 wt           -3.88     0.633       -6.13 1.12e- 6  -5.17     -2.58
## 3 hp           -0.0318   0.00903     -3.52 1.45e- 3  -0.0502   -0.0133
```

Interpreting coefficients in multiple regression (critical)

- The coefficient of `wt` in `m2` means: **"Expected change in mpg for a 1-unit increase in weight, *holding hp constant*."**
- The coefficient of `hp` in `m2` means: **"Expected change in mpg for a 1-unit increase in horsepower, *holding wt constant*."**

This "holding other variables constant" is what makes multiple regression powerful, and what makes interpretation different from simple regression.

Compare nested models (does adding hp improve the model?)

`m1` is nested within `m2` (m2 adds hp). We can use ANOVA (F-test) to compare:

```r
anova(m1, m2)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ wt
## Model 2: mpg ~ wt + hp
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1     30 278.32
## 2     29 195.05  1    83.274 12.381 0.001451 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpretation:

- If p-value is small, adding `hp` significantly improves model fit (beyond wt alone).

AIC comparison (optional model selection tool)

Lower AIC is better (penalizes complexity):

```r
AIC(m1, m2)
```

```
##    df      AIC
## m1  3 166.0294
## m2  4 156.6523
```

Regression assumptions: how to check them in R

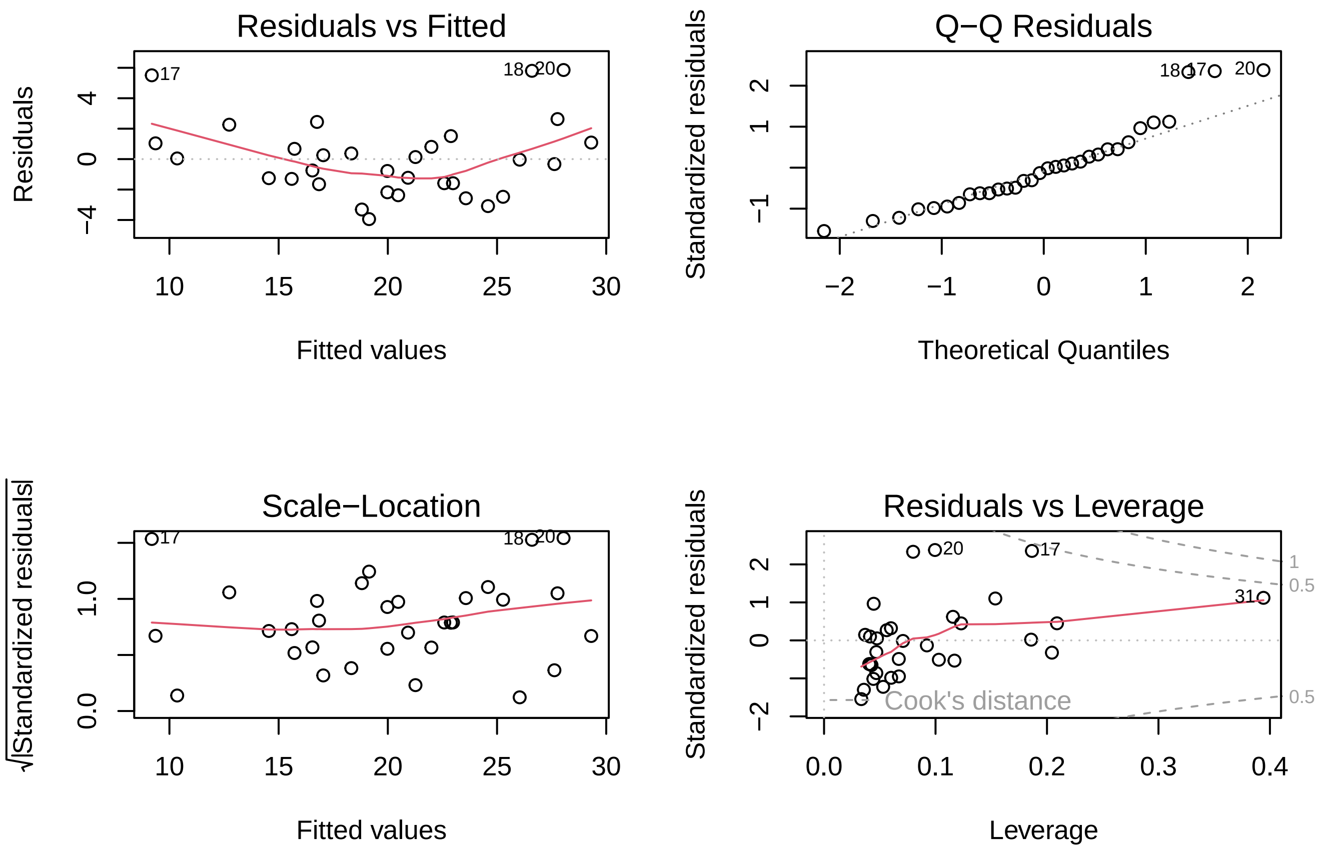
A strong regression workflow is:

1. Fit the model
2. Inspect diagnostics
3. Identify violations
4. Adjust model or interpret cautiously

We'll use `m2` (multiple regression) for diagnostics.

The "big four" diagnostic plots

```r
par(mfrow = c(2, 2))
plot(m2)
```



```
par(mfrow = c(1, 1))
```

These correspond to:

1. Residuals vs Fitted (linearity + mean-zero errors)
2. Normal Q-Q (normality of residuals)
3. Scale-Location (homoscedasticity)
4. Residuals vs Leverage (influential points)

We will now go one-by-one.

Linearity

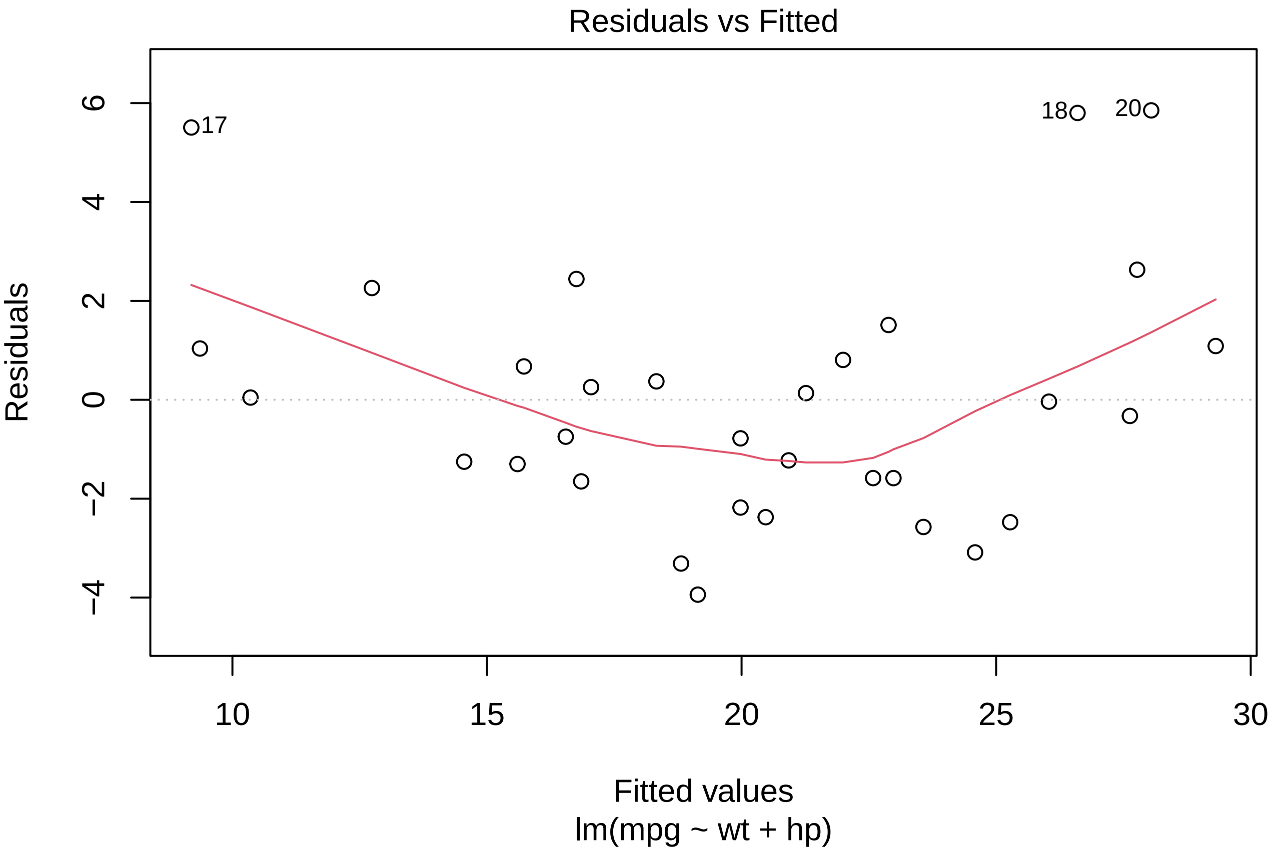
What it means

The mean of Y should be a linear function of predictors (in the parameters). If the relationship is curved, a straight-line model is misspecified.

How to check

- Residuals vs fitted should show no systematic pattern (no curve).

```
plot(m2, which = 1)
```



What to say:

- If you see a clear curve or structure, linearity is likely violated.
- Potential responses: transform variables, add polynomial terms, or use splines (beyond today's scope, but good to mention as next steps).

Independence

What it means

Residuals should be independent across observations.

When it matters

- Time series or longitudinal data: consecutive observations may be correlated.
- Clustered data: students within the same class, patients within the same hospital.

How to check (conceptual here)

With `mtcars`, independence is mostly a design assumption. In real studies, you check data collection design. For time-ordered data you can inspect residuals over time; for clustered data you may need mixed models.

(We won't run a formal autocorrelation test here, but we note the assumption and when it can fail.)

Homoscedasticity (constant variance)

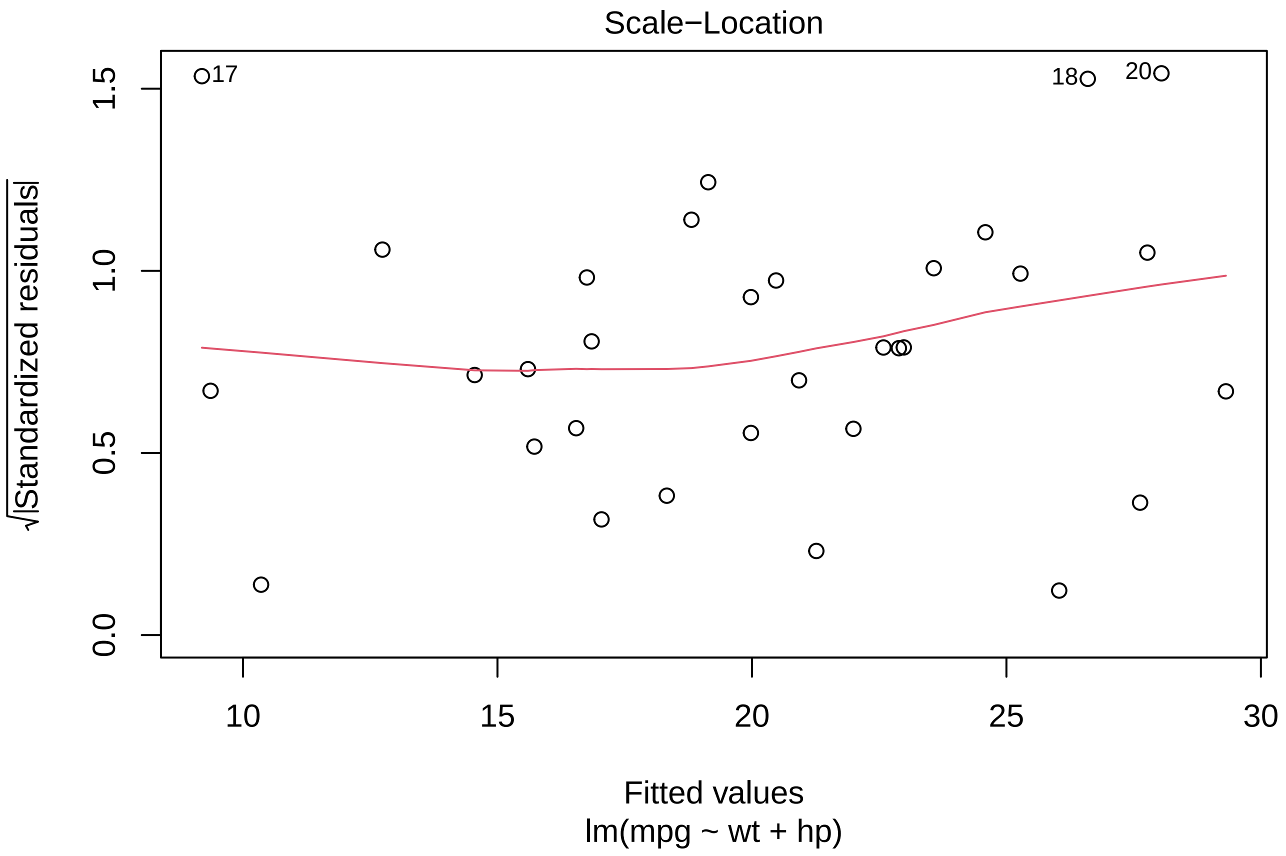
What it means

The spread of residuals should be roughly constant across fitted values.

How to check

- Look for “funnel” or “megaphone” shapes.
- Use Scale-Location plot.

```
plot(m2, which = 3)
```



If violated:

- Standard errors may be biased.
- Potential responses: transformations (e.g., log), robust standard errors, or modeling variance explicitly.

Normality of residuals

What it means

Residuals are approximately normally distributed.

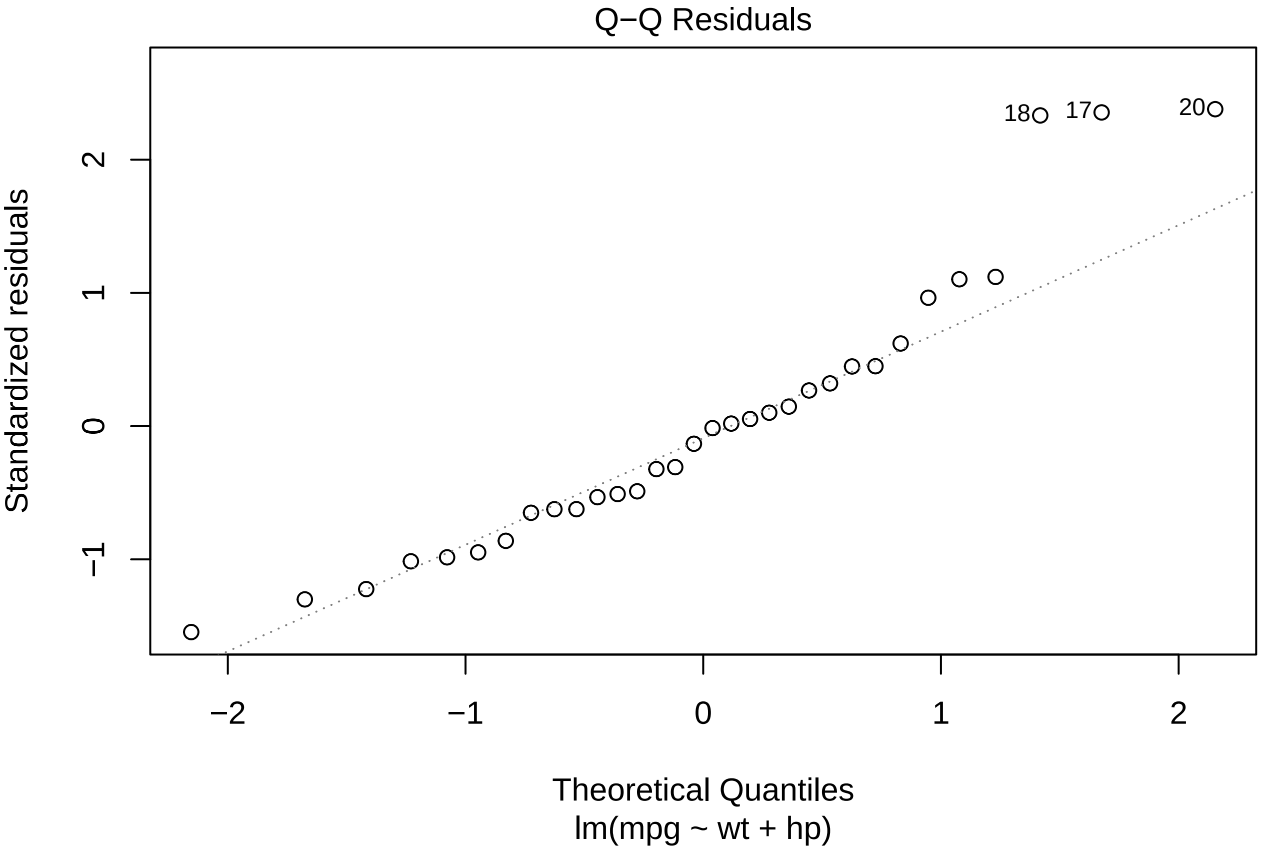
This matters mostly for:

- small-sample inference on coefficients
- confidence intervals and p-values (less critical in large n due to CLT)

How to check

- Q-Q plot: points should roughly follow the line.

```
plot(m2, which = 2)
```



Optional: Shapiro-Wilk test *(remember: can be too sensitive in large n and too weak in small n; prefer plots + context).*

```
shapiro.test(resid(m2))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(m2)
## W = 0.92792, p-value = 0.03427
```

Outliers and influential observations

Concepts (important distinctions)

- **Outlier in Y:** unusual outcome value (large residual)

- **High leverage:** unusual X values (far from center in predictor space)
- **Influential point:** changes the model noticeably (often high leverage + large residual)

Cook's distance (influence)

```
cooks <- cooks.distance(m2)

# Quick look at the largest Cook's distances

sort(cooks, decreasing = TRUE)[1:6]
```

```
##         17         31         20         18         28         21
## 0.42361090 0.27203975 0.20839326 0.15742629 0.07353985 0.02791982
```

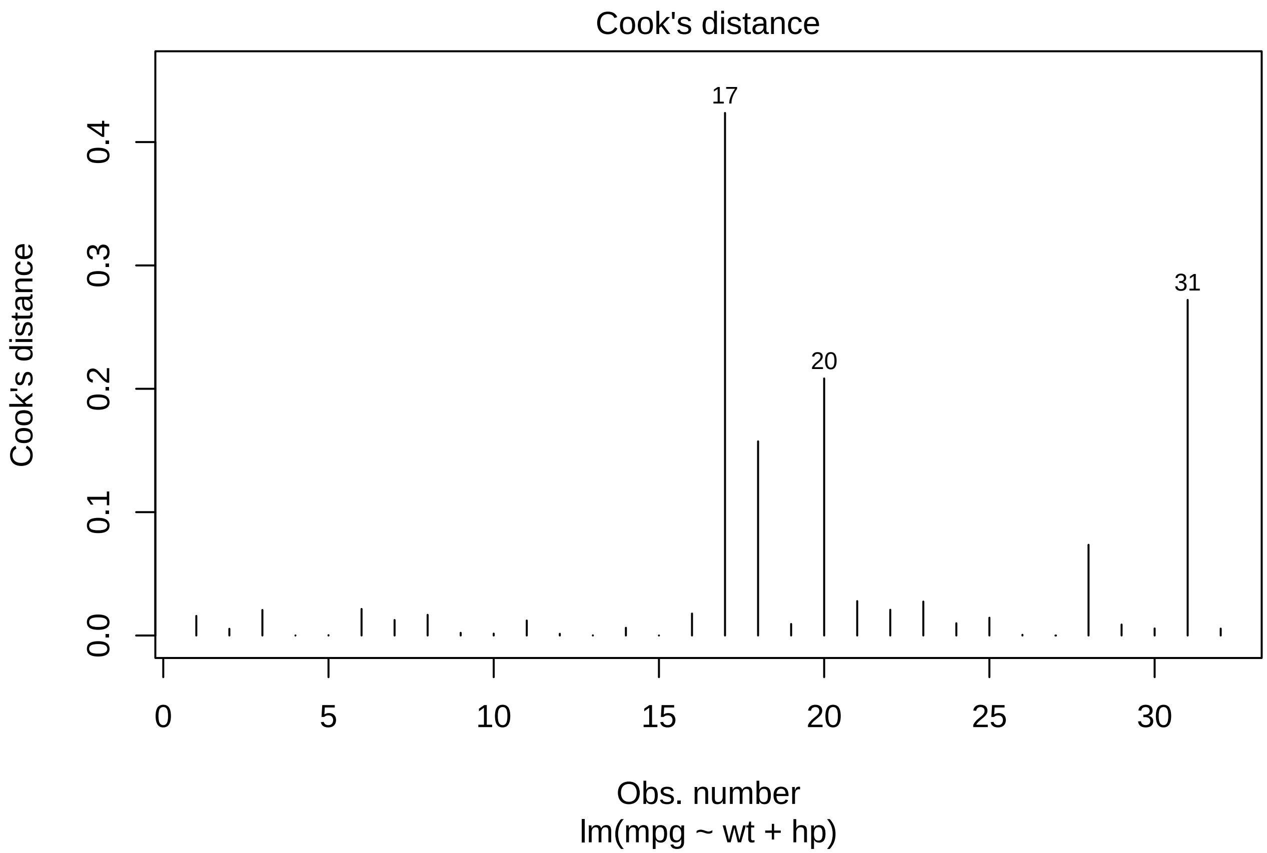
A common rule-of-thumb threshold:

$Di > 4/n$

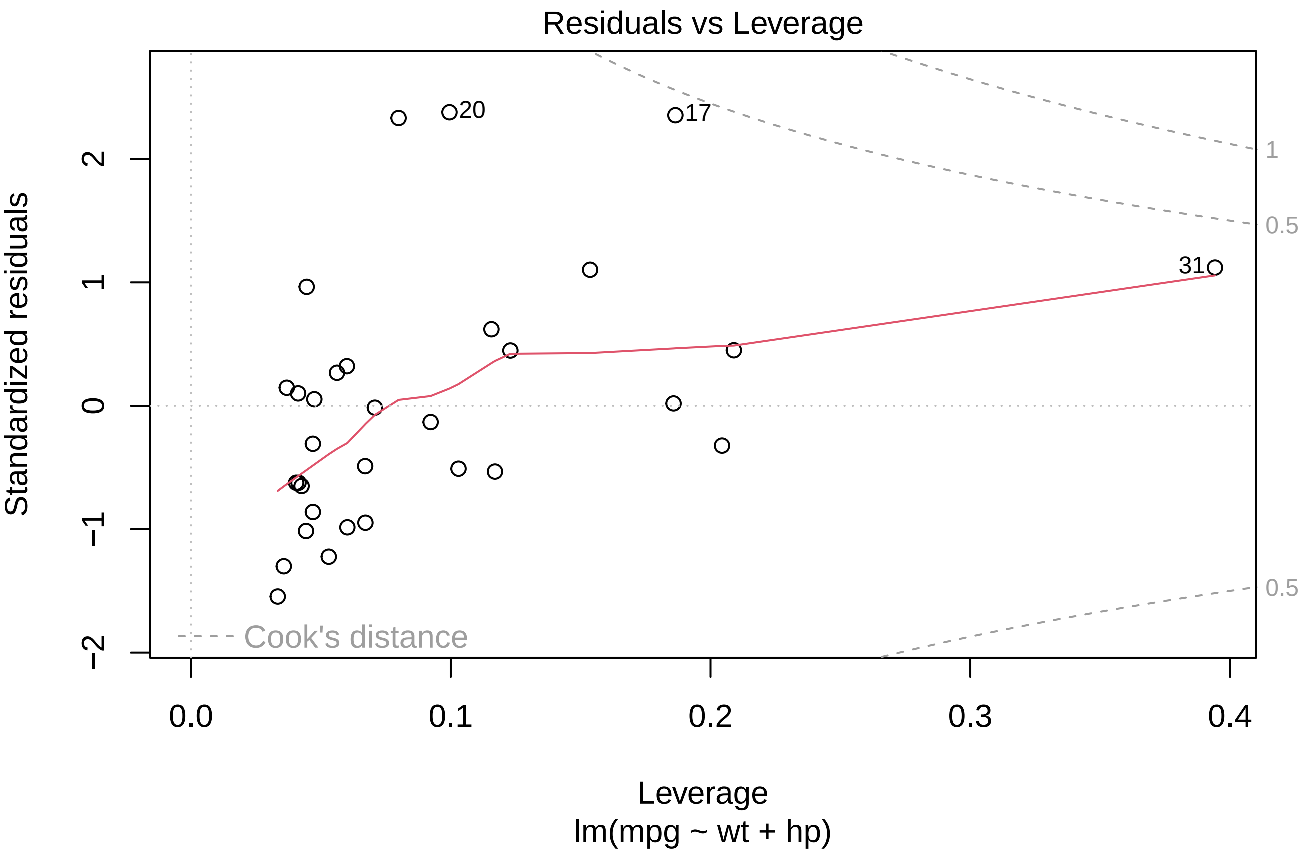
```
n <- nrow(df)
which(cooks > (4 / n))
```

```
## 17 18 20 31
## 17 18 20 31
```

```
# Plot Cook’s distance:
plot(m2, which = 4)
```



```r
# Leverage and influence plot
plot(m2, which = 5)
```



How to handle influential points (what to teach):

- Do not delete automatically.
- Investigate: data entry error? special case? legitimate observation?
- Report sensitivity: fit model with/without the point and compare conclusions.

Multicollinearity (multiple regression only)

What it means

Predictors are correlated with each other, making coefficient estimates unstable (large standard errors, sign flips).

How to check

Variance Inflation Factor (VIF):

```
vif(m2)
```

```
##       wt       hp
## 1.766625 1.766625
```

Interpretation:

- Larger VIF indicates more multicollinearity.

- There is no single universal cutoff, but very high VIF suggests interpretation problems and unstable estimates.

Model refinement examples (guided options)

This section shows how you might *improve* a model when diagnostics indicate issues.

Add an interaction term (optional extension)

Sometimes the effect of weight depends on horsepower:

$mpg = b_0\ + b_1\ wt + b_2\ hp + b_3\ (wt \cdot hp) + \epsilon$

```
m3 <- lm(mpg ~ wt * hp, data = df)
summary(m3)
```

```
##
## Call:
## lm(formula = mpg ~ wt * hp, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0632 -1.6491 -0.7362  1.4211  4.5513
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 49.80842    3.60516  13.816 5.01e-14 ***
## wt          -8.21662    1.26971  -6.471 5.20e-07 ***
## hp          -0.12010    0.02470  -4.863 4.04e-05 ***
## wt:hp        0.02785    0.00742   3.753 0.000811 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.153 on 28 degrees of freedom
## Multiple R-squared:  0.8848, Adjusted R-squared:  0.8724
## F-statistic: 71.66 on 3 and 28 DF,  p-value: 2.981e-13
```

```
anova(m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ wt + hp
## Model 2: mpg ~ wt * hp
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     29 195.05
## 2     28 129.76  1    65.286 14.088 0.0008108 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Interpretation:

- If interaction is significant, the slope of `wt` differs across levels of `hp` (or vice versa). Interpretation becomes conditional.

Centering predictors (helps interpretation; sometimes helps collinearity)

Centering makes the intercept meaningful at the *average* predictor values.

```
df_centered <- df %>%
mutate(
wt_c = wt - mean(wt),
hp_c = hp - mean(hp)
)

m2c <- lm(mpg ~ wt_c + hp_c, data = df_centered)
summary(m2c)
```

```
##
## Call:
## lm(formula = mpg ~ wt_c + hp_c, data = df_centered)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.941 -1.600 -0.182  1.050  5.854
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20.09062    0.45846  43.822  < 2e-16 ***
## wt_c        -3.87783    0.63273  -6.129 1.12e-06 ***
## hp_c        -0.03177    0.00903  -3.519  0.00145 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.593 on 29 degrees of freedom
## Multiple R-squared:  0.8268, Adjusted R-squared:  0.8148
## F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12
```

```
tidy(m2c, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)  20.1      0.458       43.8  4.69e-28  19.2      21.0
## 2 wt_c         -3.88     0.633       -6.13 1.12e- 6  -5.17     -2.58
## 3 hp_c         -0.0318   0.00903     -3.52 1.45e- 3  -0.0502   -0.0133
```

Logistic Regression (Categorical Outcomes)

When linear regression is not appropriate

Linear regression assumes a continuous outcome variable. When the outcome is binary or categorical (e.g., yes/no, pass/fail), linear regression is not suitable.

Logistic regression models the probability of an outcome instead.

Relationship to the chi-square test

- Chi-square asks: Is there an association?
- Logistic regression asks: How does each predictor change the probability?

Logistic regression can be seen as a model-based extension of the chi-square test that allows multiple predictors and effect size estimation.

Reporting regression results (templates)

Simple regression report template

"Weight significantly predicted fuel efficiency, $b = ..., t(df) = ..., p = ..., R^2 = ...$"

Multiple regression report template

"A multiple linear regression was fit to predict mpg from weight and horsepower.
The model explained $R^2 = ...$ of variance in mpg (Adj. $R^2 = ...$). Holding the other predictor constant, weight was associated with a change of … mpg per unit, and horsepower was associated with a change of … mpg per unit."

You can extract the values for reporting:

```
coefs <- tidy(m2, conf.int = TRUE)
fit   <- glance(m2)

coefs
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)  37.2      1.60        23.3  2.57e-20  34.0      40.5
## 2 wt           -3.88     0.633       -6.13 1.12e- 6  -5.17     -2.58
## 3 hp           -0.0318   0.00903     -3.52 1.45e- 3  -0.0502   -0.0133
```

```
fit %>% select(r.squared, adj.r.squared, sigma, statistic, p.value, df.residual)
```

```
## # A tibble: 1 x 6
##   r.squared adj.r.squared sigma statistic  p.value df.residual
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl>       <int>
## 1     0.827         0.815  2.59      69.2 9.11e-12          29
```

Short exercises

Exercise 1: Correlation (quick)

1. Use `cor.test()` to compute Pearson and Spearman correlation between `wt` and `mpg`.
2. Compare results and explain any differences.

```r
cor.test(df$wt, df$mpg, method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$wt and df$mpg
## t = -9.559, df = 30, p-value = 1.294e-10
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.9338264 -0.7440872
## sample estimates:
##        cor
## -0.8676594
```

```r
cor.test(df$wt, df$mpg, method = "spearman", exact = FALSE)
```

```
##
##  Spearman's rank correlation rho
##
## data:  df$wt and df$mpg
## S = 10292, p-value = 1.488e-11
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## -0.886422
```

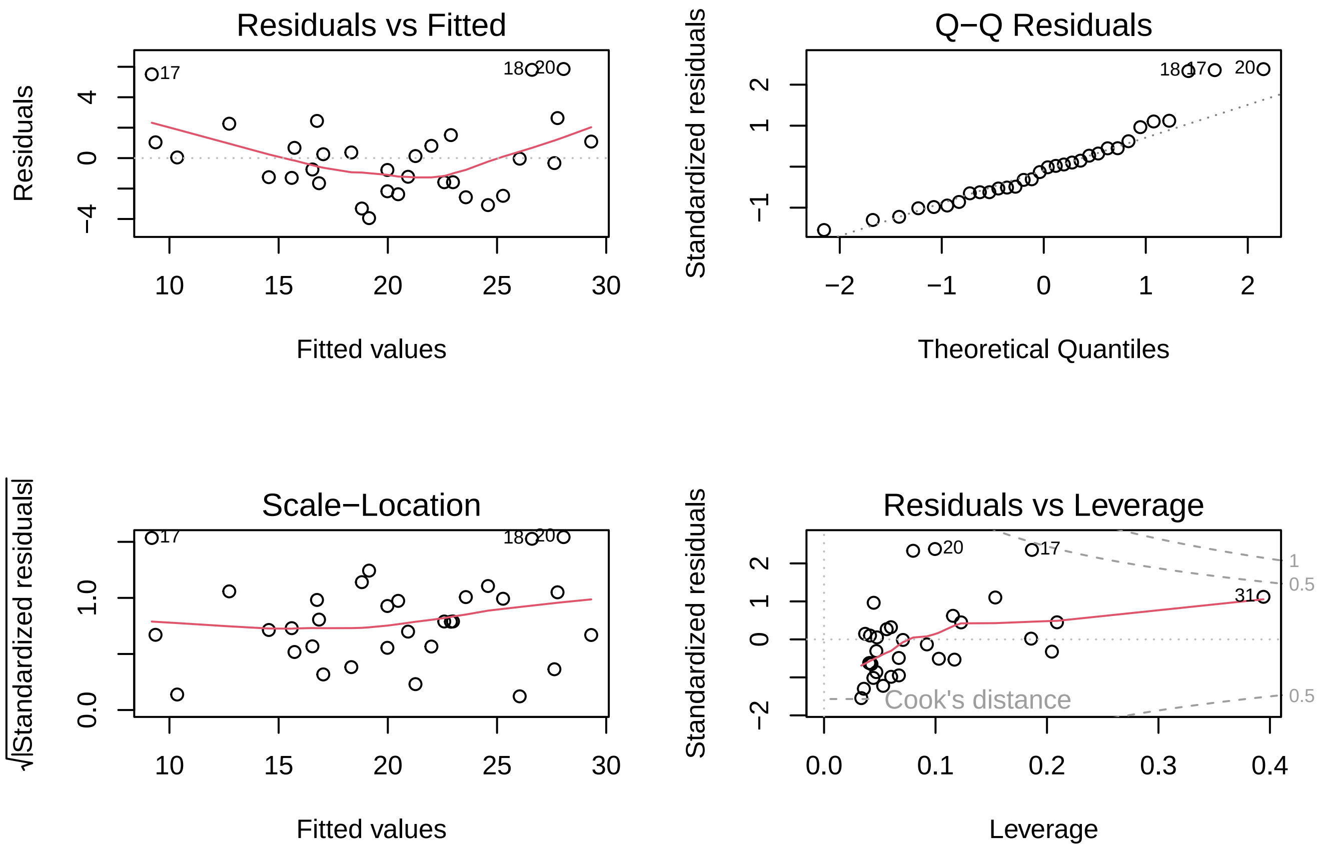
Exercise 2: Build and diagnose your own regression

1. Fit `mpg ~ wt`
2. Fit `mpg ~ wt + hp`
3. Compare models using `anova(m1, m2)`
4. Inspect the 4 diagnostic plots for `m2` and write 2–3 sentences:
   - Is linearity plausible?
   - Any signs of heteroscedasticity?
   - Any influential points?

```r
m1_ex <- lm(mpg ~ wt, data = df)
m2_ex <- lm(mpg ~ wt + hp, data = df)

anova(m1_ex, m2_ex)
```

```
## Analysis of Variance Table
##
## Model 1: mpg ~ wt
## Model 2: mpg ~ wt + hp
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1     30 278.32
## 2     29 195.05  1    83.274 12.381 0.001451 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
par(mfrow = c(2, 2))
plot(m2_ex)
```



```r
par(mfrow = c(1, 1))
```

Appendix — Cheatsheet (what you should remember)

Correlation

- `cor(x, y, method="pearson"|"spearman"|"kendall")`
- `cor.test(x, y, ...)` for inference + CI

Regression basics

- `lm(y ~ x, data=df)`
- `summary(model)` for coefficients, R^2, tests
- `tidy(model)` and `glance(model)` for clean tables

Diagnostics

- `plot(model)` (4 key diagnostic plots)
- `cooks.distance(model)` for influence
- `car::vif(model)` for multicollinearity

  `shapiro.test(resid(model))` (use plots + context)