DSA 8020 R Session 1: Simple Linear Regression

Whitney Huang, Clemson University

Contents

Example: Maximum Heart Rate vs. Age

The maximum heart rate (HR_{max}) of a person is often said to be related to age (Age) by the following equation:

$$\text{HR}_{max} = 220 - \text{Age}$$

Let's use a dataset to assess the validity of this statement.

Load the dataset

There are several ways to load a dataset into R; for example, one could import the data over the Internet

```
# read.csv() -> Reads a CSV file in table format and creates a data frame from it
dat <- read.csv('http://whitneyhuang83.github.io/STAT8010/Data/maxHeartRate.csv', header = T)
# header = TRUE -> Indicates that the first row contains column names

# Preview the dataset
# head() displays the first 6 rows so we can quickly inspect the structure
head(dat)
```

```
##   Age MaxHeartRate
## 1  18          202
## 2  23          186
## 3  25          187
## 4  35          180
## 5  65          156
## 6  54          169
```

Summarize the data before fitting models

```
# Extract response variable (Max Heart Rate) and predictor (Age)
y <- dat$MaxHeartRate; x <- dat$Age

# Display summary statistics (min, quartiles, median, mean, max)
summary(dat)
```

```
##       Age         MaxHeartRate
##  Min.   :18.00   Min.   :153.0
##  1st Qu.:23.00   1st Qu.:173.0
##  Median :35.00   Median :180.0
##  Mean   :37.33   Mean   :180.3
##  3rd Qu.:48.00   3rd Qu.:190.0
##  Max.   :72.00   Max.   :202.0
```

```
# Compute variability of each variable
var(x); var(y)
```

```
## [1] 305.8095
```

```
## [1] 214.0667
```

```
# Measure joint variability between Age and Max Heart Rate
cov(x, y) # Covariance (direction of linear relationship)
```

```
## [1] -243.9524
```

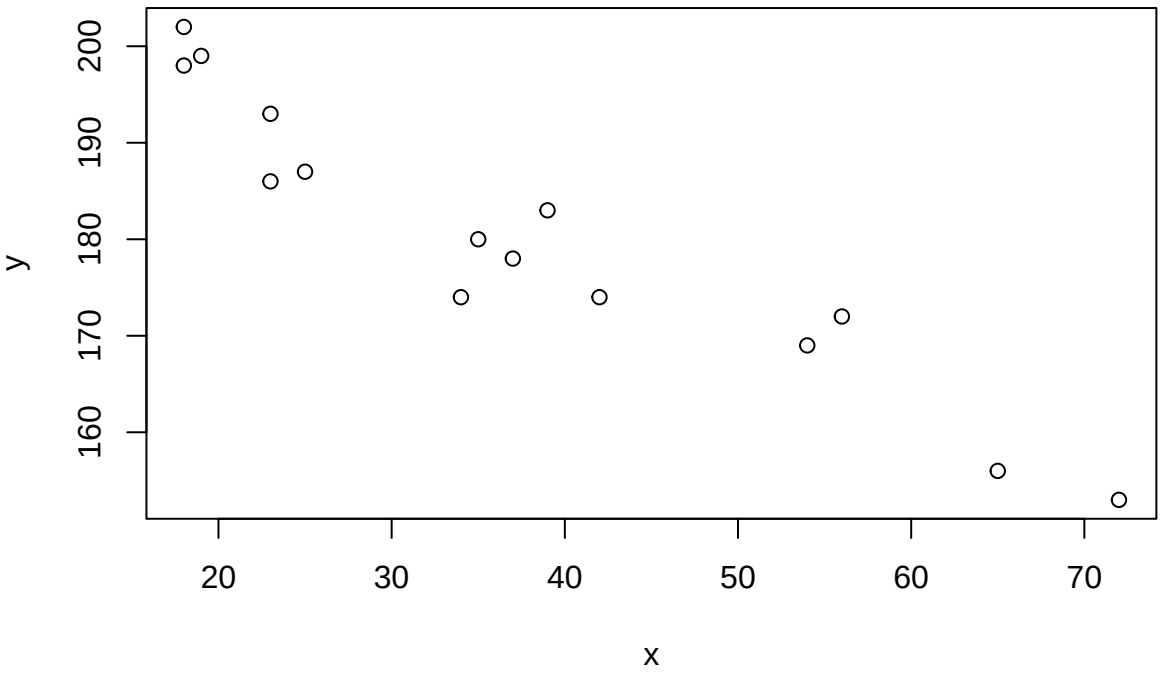
```
# Measure strength and direction of linear association (scaled)
cor(x, y) # Correlation (ranges from -1 to 1)
```

```
## [1] -0.9534656
```

Plot the data before fitting models

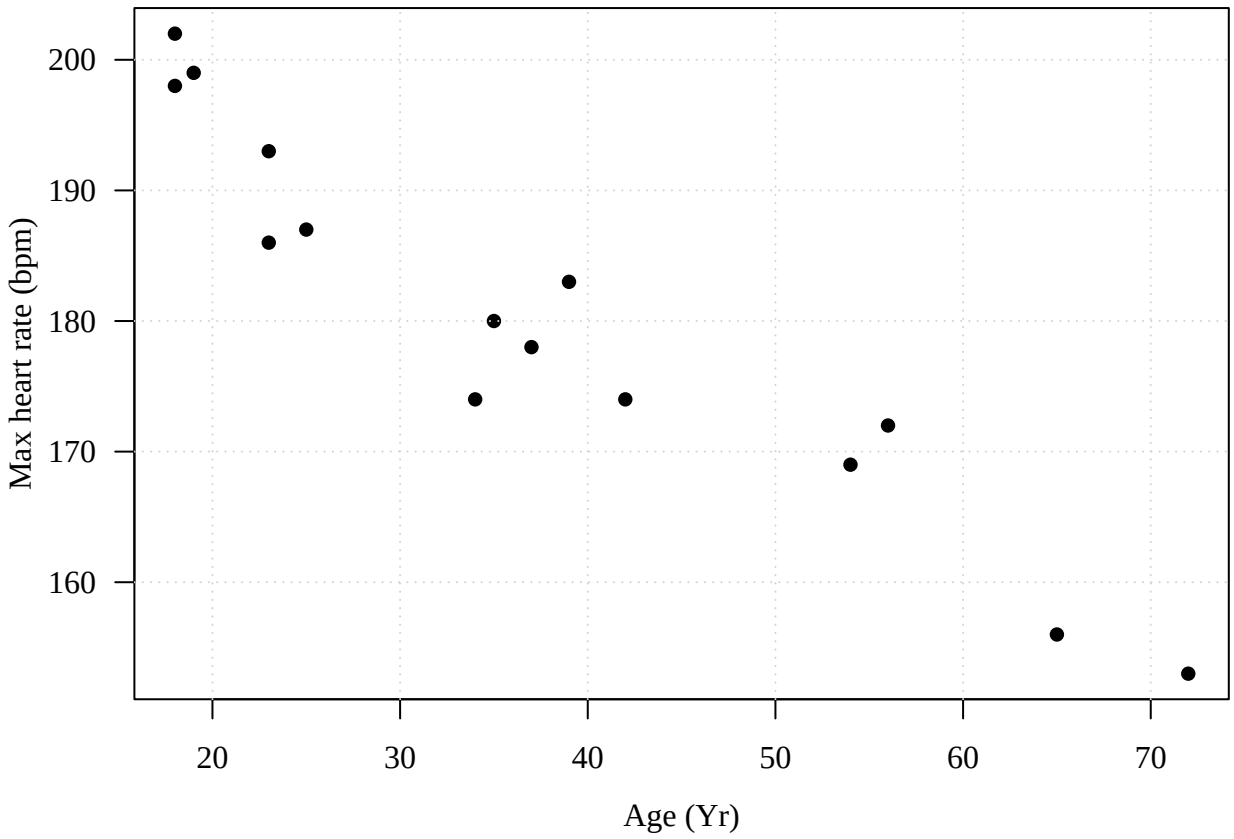
This is what the `scatterplot` would look like by default. Place the predictor (`age`) as the first argument and the response (`maxHeartRate`) as the second argument.

```
plot(x, y)
```



Let's make the plot look nicer (type `?plot` to learn more).

```
# Set graphical parameters:
# las = 1 → horizontal axis labels
# mar → margins (bottom, left, top, right)
# mgp → spacing for axis title, labels, and line
# family = "serif" → font style
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(x, y, pch = 16, xlab = "Age (Yr)", ylab = "Max heart rate (bpm)")
# Add grid lines for better readability
grid()
```



Simple Linear Regression

Estimation

Let's perform the calculations to determine the regression coefficients as well as the standard deviation of the random error.

Slope: $\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$

```
# Compute components needed for slope estimation
# Center variables by subtracting their means
y_diff <- y - mean(y) # Deviations of response from mean
x_diff <- x - mean(x) # Deviations of predictor from mean

# Estimate slope (beta_1) using least squares formula
beta_1 <- sum(y_diff * x_diff) / sum((x_diff)^2)
beta_1
```

```
## [1] -0.7977266
```

Intercept: $\hat{\beta}_0 = \bar{y} - \bar{x}\hat{\beta}_1$

```
# Estimate intercept (beta_0)
# Ensures regression line passes through (mean(x), mean(y))
beta_0 <- mean(y) - mean(x) * beta_1
beta_0
```

```
## [1] 210.0485
```

Fitted values: $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$

```
# Compute fitted values (predicted responses)
y_hat <- beta_0 + beta_1 * x
y_hat
```

```
##  [1] 195.6894 191.7007 190.1053 182.1280 158.1962 166.9712 182.9258 165.3758
##  [9] 152.6121 194.8917 191.7007 176.5439 195.6894 178.9371 180.5326
```

$\hat{\sigma}$**:** $\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}$

```
# Estimate variance of random error (sigma^2)
# Uses residual sum of squares divided by (n - 2) degrees of freedom
sigma2 <- sum((y - y_hat)^2) / (length(y) - 2)

# Standard deviation of error (sigma)
sqrt(sigma2)
```

```
## [1] 4.577799
```

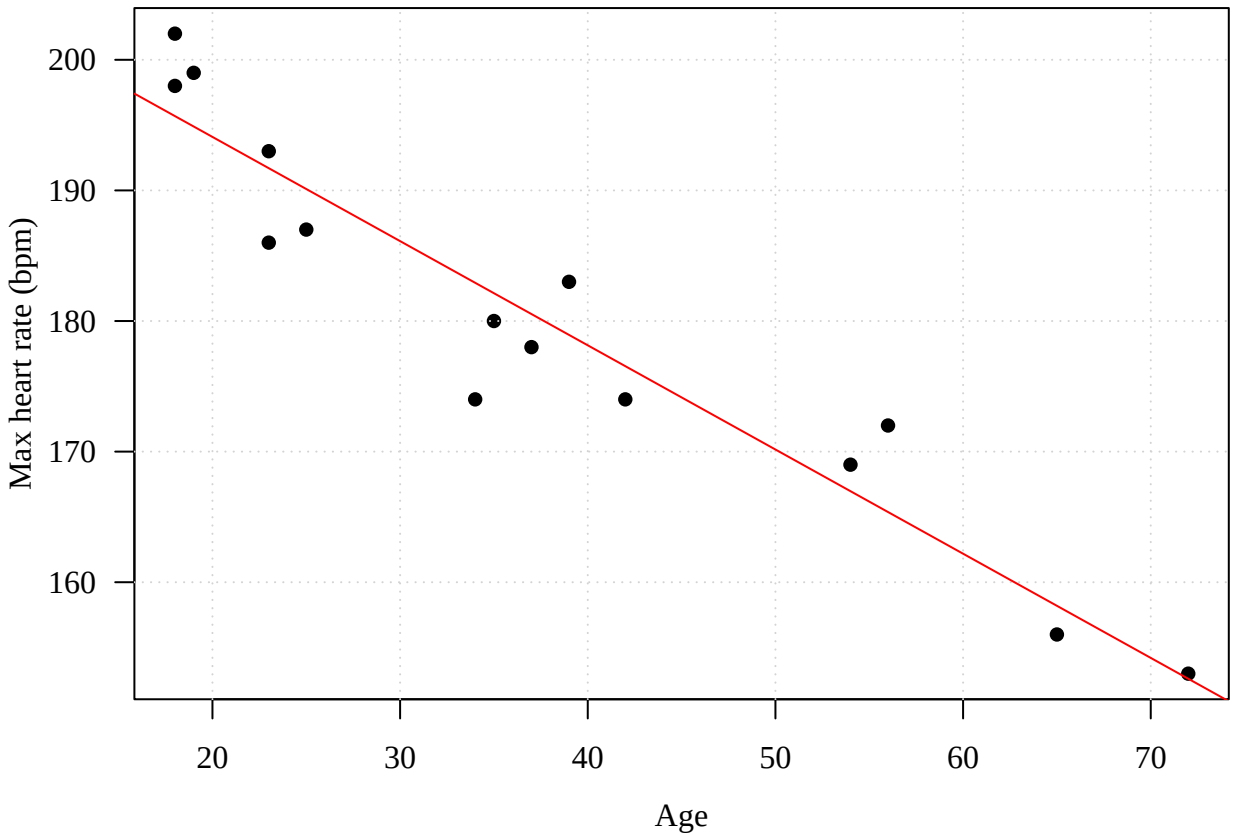
Add the fitted regression line to the `scatterplot`

```
# Visualize data with fitted regression line

# Set plotting parameters
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")

# Scatterplot of data
plot(x, y, pch = 16, xlab = "Age", ylab = "Max heart rate (bpm)")
grid()

# Add fitted regression line
abline(a = beta_0, b = beta_1, col = "red")
```



Let R do all the work

```
# Fit the same model using built-in R function lm()
fit <- lm(MaxHeartRate ~ Age, data = dat)

# Display model summary:
# - Coefficient estimates (beta_0, beta_1)
# - Standard errors
# - R-squared and significance tests
summary(fit)
```

```
##
## Call:
## lm(formula = MaxHeartRate ~ Age, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9258 -2.5383  0.3879  3.1867  6.6242
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 210.04846    2.86694   73.27  < 2e-16 ***
## Age          -0.79773    0.06996  -11.40 3.85e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.578 on 13 degrees of freedom
```

```
## Multiple R-squared:  0.9091, Adjusted R-squared:  0.9021
## F-statistic:   130 on 1 and 13 DF,  p-value: 3.848e-08
```

- Regression coefficients

```
fit$coefficients
```

```
## (Intercept)         Age
## 210.0484584  -0.7977266
```

- Fitted values

```
fit$fitted.values
```

```
##        1        2        3        4        5        6        7        8
## 195.6894 191.7007 190.1053 182.1280 158.1962 166.9712 182.9258 165.3758
##        9       10       11       12       13       14       15
## 152.6121 194.8917 191.7007 176.5439 195.6894 178.9371 180.5326
```

- $\hat{\sigma}$

```
summary(fit)$sigma
```

```
## [1] 4.577799
```

Model Checking: Residual Analysis

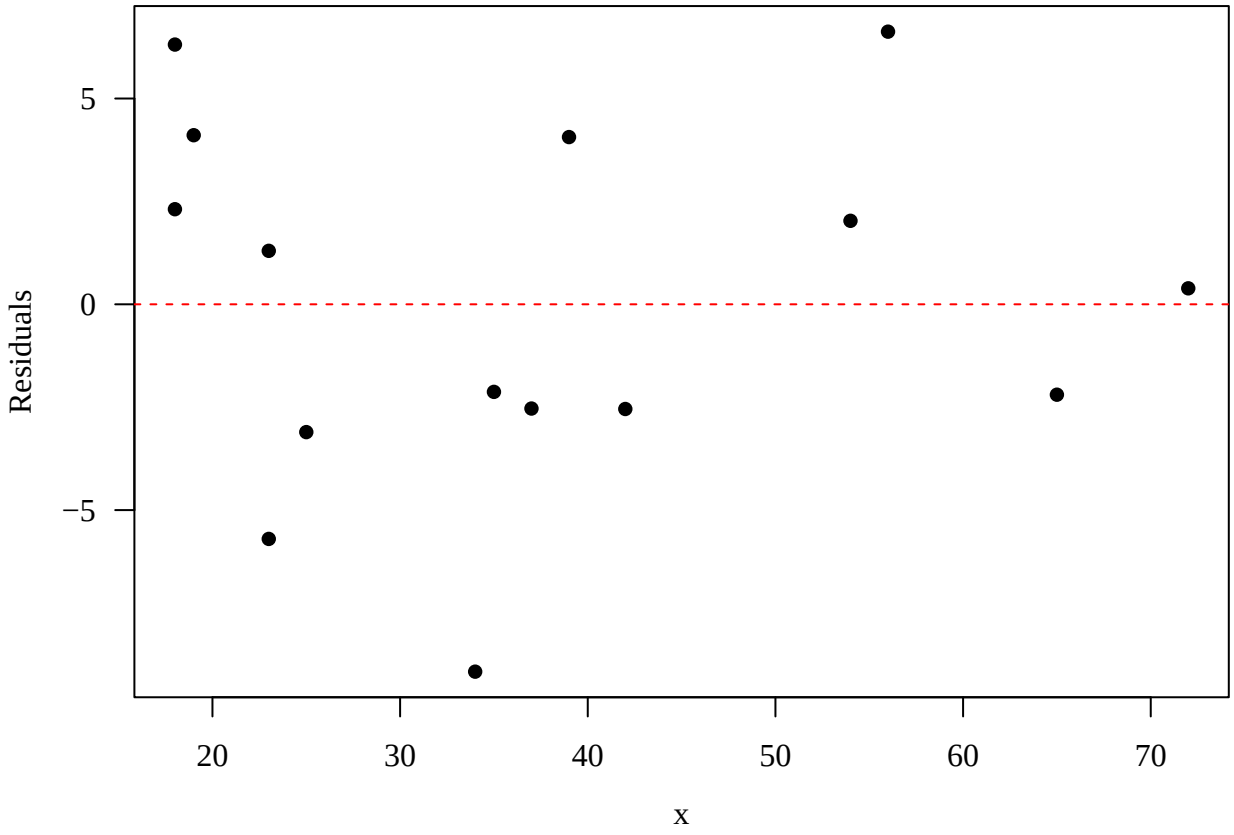
Assumptions on error ε:

- $\mathrm{E}[\varepsilon_i] = 0$
- $\mathrm{Var}[\varepsilon_i] = \sigma^2$
- $\varepsilon_i \overset{iid}{\sim} \mathrm{N}(0, \sigma^2)$

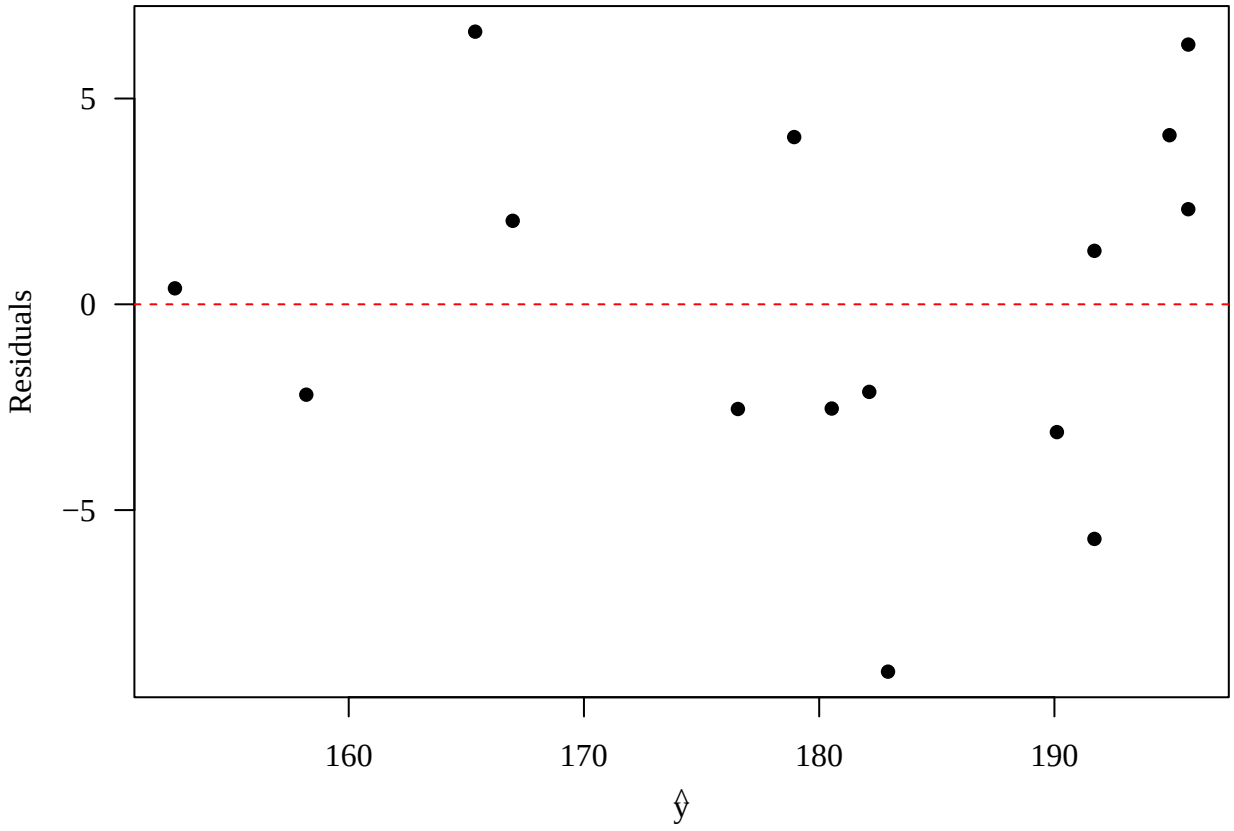
We use $e_i = y_i - \hat{y}_i$, where $i = 1, \cdots, n$ to assess these model assumptions.

Residual plots

```
## Residuals vs. predictor (x)
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(x, fit$residuals, pch = 16, ylab = "Residuals")
# Add horizontal reference line at 0
abline(h = 0, col = "red", lty = 2)
```

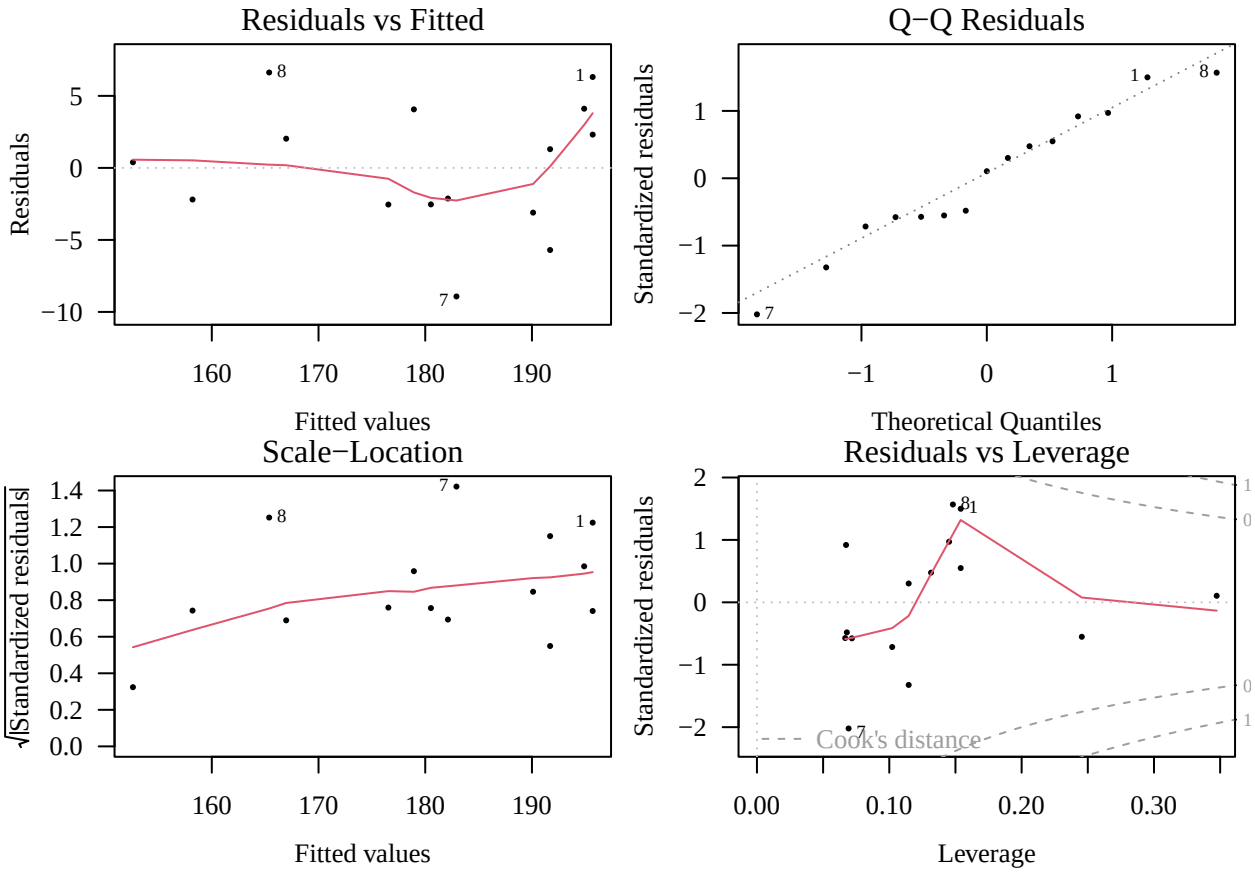



```r
## Residuals vs. fitted values (y_hat)
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(fit$fitted.values, fit$residuals, pch = 16, ylab = "Residuals", xlab = expression(hat(y)))
# Add horizontal reference line at 0
abline(h = 0, col = "red", lty = 2)
```



Plot Diagnostics for an `lm` Object

```
# Set plotting layout:
# mfrow = c(2, 2) → arrange plots in a 2x2 grid
par(las = 1, mfrow = c(2, 2), mar = c(3.5, 3.5, 1.25, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(fit, cex = 0.5, pch = 16)
```



(1) Residuals vs Fitted
- **Purpose:** Check linearity and constant variance
- **What you want:** Random scatter around 0
- **Red flags:**
- Curve → nonlinearity
- Funnel shape → heteroscedasticity

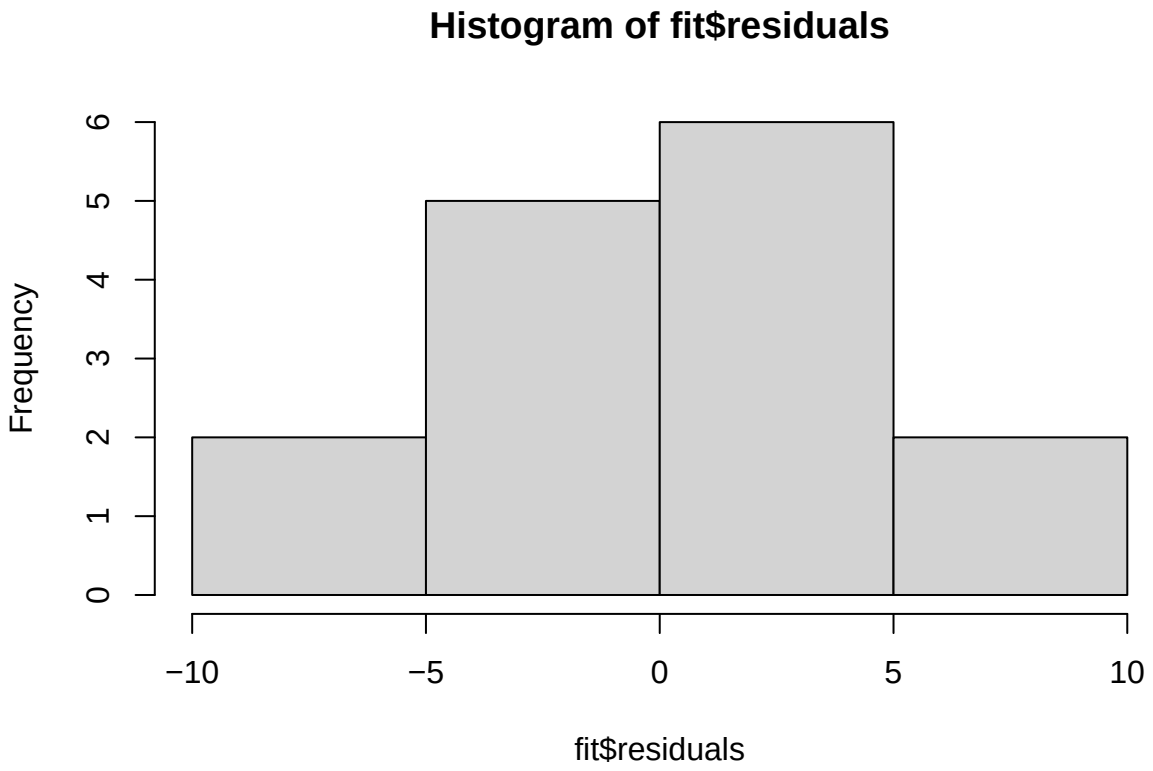
(2) Normal Q-Q Plot
- **Purpose:** Check if residuals are approximately normal
- **What you want:** Points lie on the straight line
- **Red flags:**
- Systematic deviation → non-normal errors
- Heavy tails → outliers

(3) Scale-Location (Spread vs Fitted)
- **Purpose:** Check homoscedasticity (constant variance)
- **What you want:** Flat horizontal pattern
- **Red flags:**
- Increasing spread → variance increases with fitted values

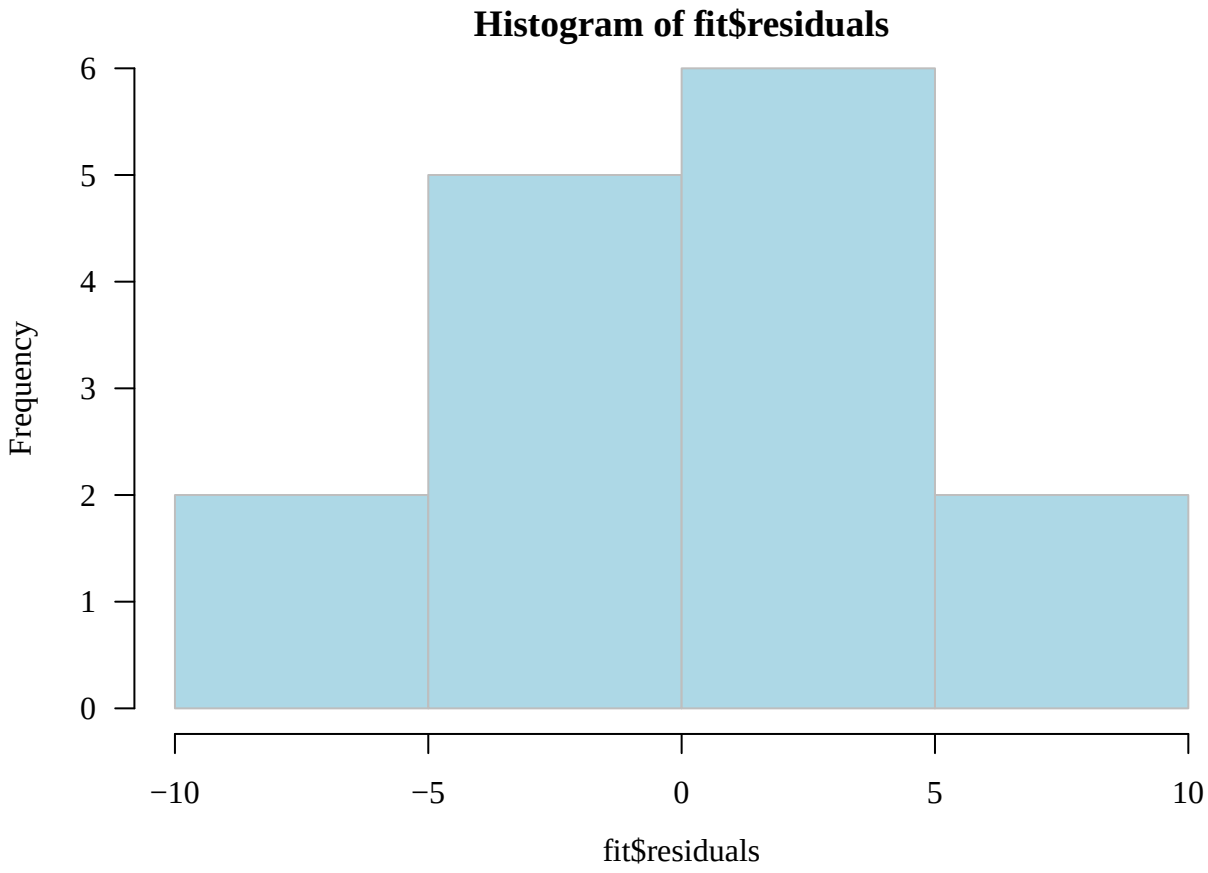
(4) Residuals vs Leverage
- **Purpose:** Identify influential observations
- **What you want:** No points with both high leverage and large residuals
- **Red flags:**
- Points outside Cook's distance lines → influential points

Assessing normality of random error

```r
# Histogram of residuals (basic)
# Provides a quick visual check of the distribution shape
hist(fit$residuals)
```

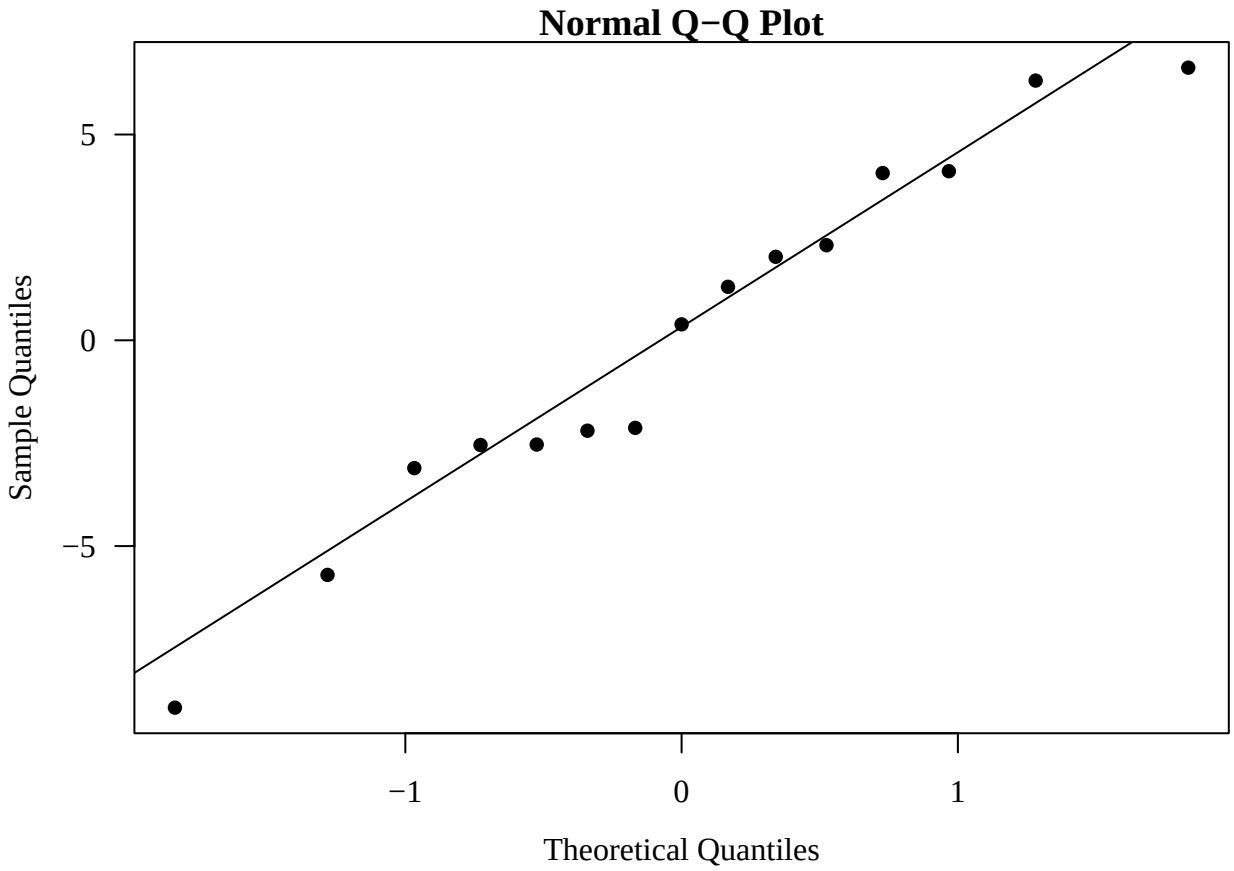



```r
# Improved histogram with styling: col → fill color, border → bar outline
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
hist(fit$residuals, 5, col = "lightblue", border = "gray", las = 1)
```




```r
# Normal Q-Q plot
# Compares sample quantiles of residuals to theoretical normal quantiles
qqnorm(fit$residuals, pch = 16, las = 1)

# Add reference line for normality
# Points should lie close to this line if residuals are approximately normal
qqline(fit$residuals)
```



Statistical Inference

Confidence Intervals for β_0 and β_1

```
# Set significance level
alpha = 0.05   # 95% confidence level

# Extract estimated slope (beta_1) from model summary
beta1_hat <- summary(fit)[["coefficients"]][, 1][2]

# Extract standard error of slope estimate
se_beta1 <- summary(fit)[["coefficients"]][, 2][2]

# Manually compute confidence interval for beta_1
# Uses t-distribution with (n - 2) degrees of freedom
CI_beta1 <- c(beta1_hat - qt(1 - alpha / 2, 13) * se_beta1,
              beta1_hat + qt(1 - alpha / 2, 13) * se_beta1)
CI_beta1
```

```
##        Age        Age
## -0.9488720 -0.6465811
```

```r
# Use built-in function to compute confidence intervals for all coefficients
confint(fit)
```

```
##                  2.5 %      97.5 %
## (Intercept) 203.854813 216.2421034
## Age          -0.948872  -0.6465811
```

Confidence and prediction intervals for $\mathrm{E}[Y_{new}|x_{new} = 40]$

```r
# Create a new data point for prediction
Age_new = data.frame(Age = 40)

# Manually compute predicted mean response at Age = 40
# hat_Y = beta_0 + beta_1 * x_new
hat_Y <- fit$coefficients[1] + fit$coefficients[2] * 40
hat_Y
```

```
## (Intercept)
##    178.1394
```

```r
# Confidence interval for mean response E[Y | x = 40]
# Reflects uncertainty in estimating the regression line
predict(fit, Age_new, interval = "confidence", level = 0.95)
```

```
##        fit      lwr      upr
## 1 178.1394 175.5543 180.7245
```

```r
# Prediction interval for a new observation at x = 40
# Includes both model uncertainty and random error → wider interval
predict(fit, Age_new, interval = "predict", level = 0.95)
```

```
##        fit      lwr      upr
## 1 178.1394 167.9174 188.3614
```

Check

```r
# Estimate standard deviation of random error (sigma)
# Uses residual sum of squares divided by (n - 2)
sd <- sqrt((sum(fit$residuals^2) / 13))
n <- length(y)
# Compute margin of error (ME) for prediction interval
# Components:
#  1 → variability of new observation
#  1/n → estimation of mean
#  leverage term → distance from mean(x)
ME <- qt(1 - alpha / 2, n - 2) * sd * sqrt(1 + 1 / n + (40 - mean(x))^(2) / sum((x - mean(x))^2))
c(hat_Y - ME, hat_Y + ME)
```

```
## (Intercept) (Intercept)
##    167.9174    188.3614
```

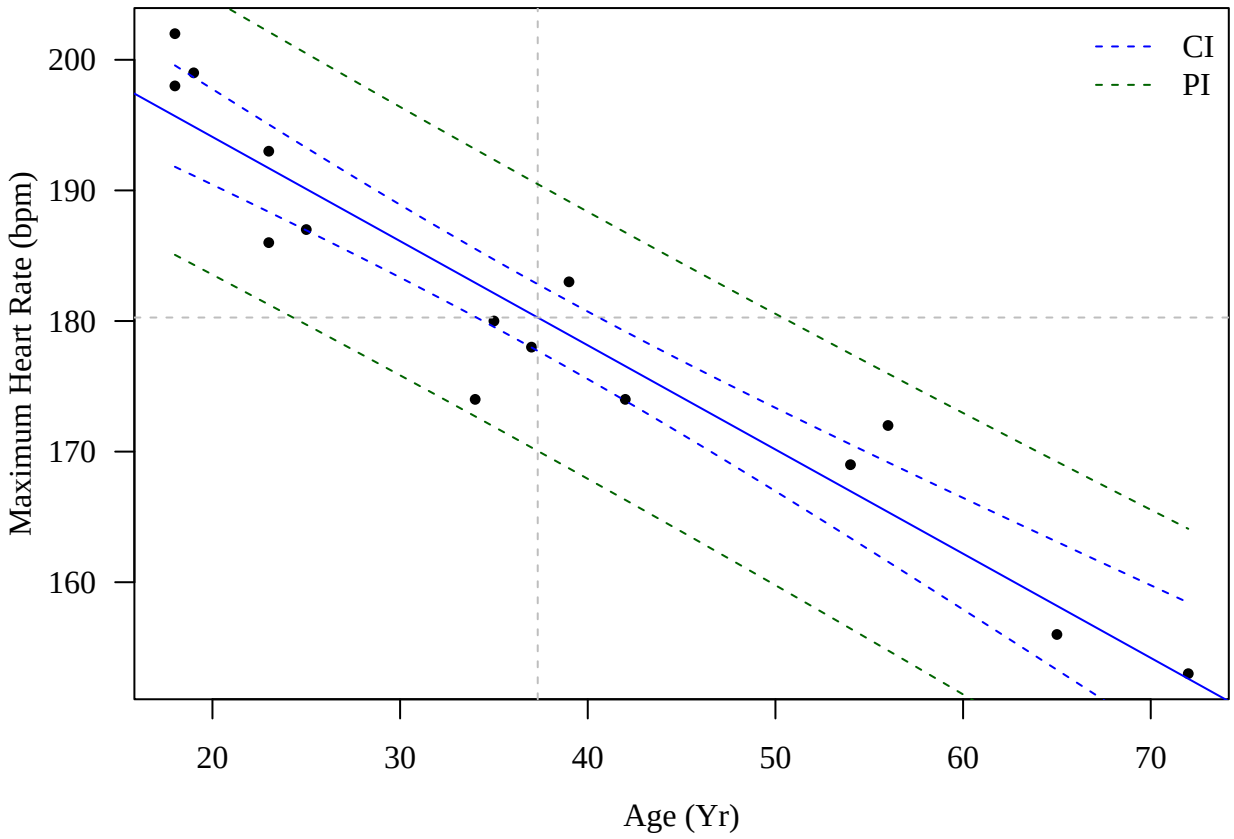
Constrcuting pointwise CIs/PIs

```r
# Create a grid of Age values for prediction (from 18 to 72)
Age_grid = data.frame(Age = 18:72)

# Compute confidence interval (CI) band for mean response
# Gives uncertainty around the regression line
CI_band <- predict(fit, Age_grid, interval = "confidence")

# Compute prediction interval (PI) band for new observations
# Includes both model uncertainty and random error → wider band
PI_band <- predict(fit, Age_grid, interval = "predict")
```

```r
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")
plot(dat$Age, dat$MaxHeartRate, pch = 16, cex = 0.75,
     xlab = "Age (Yr)", ylab = "Maximum Heart Rate (bpm)", las = 1)
# Add fitted regression line
abline(fit, col = "blue")
# Add reference lines at mean of x and y
abline(v = mean(dat$Age), lty = 2, col = "gray")
abline(h = mean(dat$MaxHeartRate), lty = 2, col = "gray")
# Add confidence interval bands (dashed blue lines)
lines(18:72, CI_band[, 2], lty = 2, col = "blue")
lines(18:72, CI_band[, 3], lty = 2, col = "blue")
# Add prediction interval bands (dashed green lines)
lines(18:72, PI_band[, 2], lty = 2, col = "darkgreen")
lines(18:72, PI_band[, 3], lty = 2, col = "darkgreen")
# Add legend to distinguish CI and PI
legend("topright", legend = c("CI", "PI"), col = c("blue", "darkgreen"), lty = 2, bty = "n")
```



Hypothesis Tests for β_1

$H_0 : \beta_1 = -1$ vs. $H_a : \beta_1 \neq -1$ with $\alpha = 0.05$

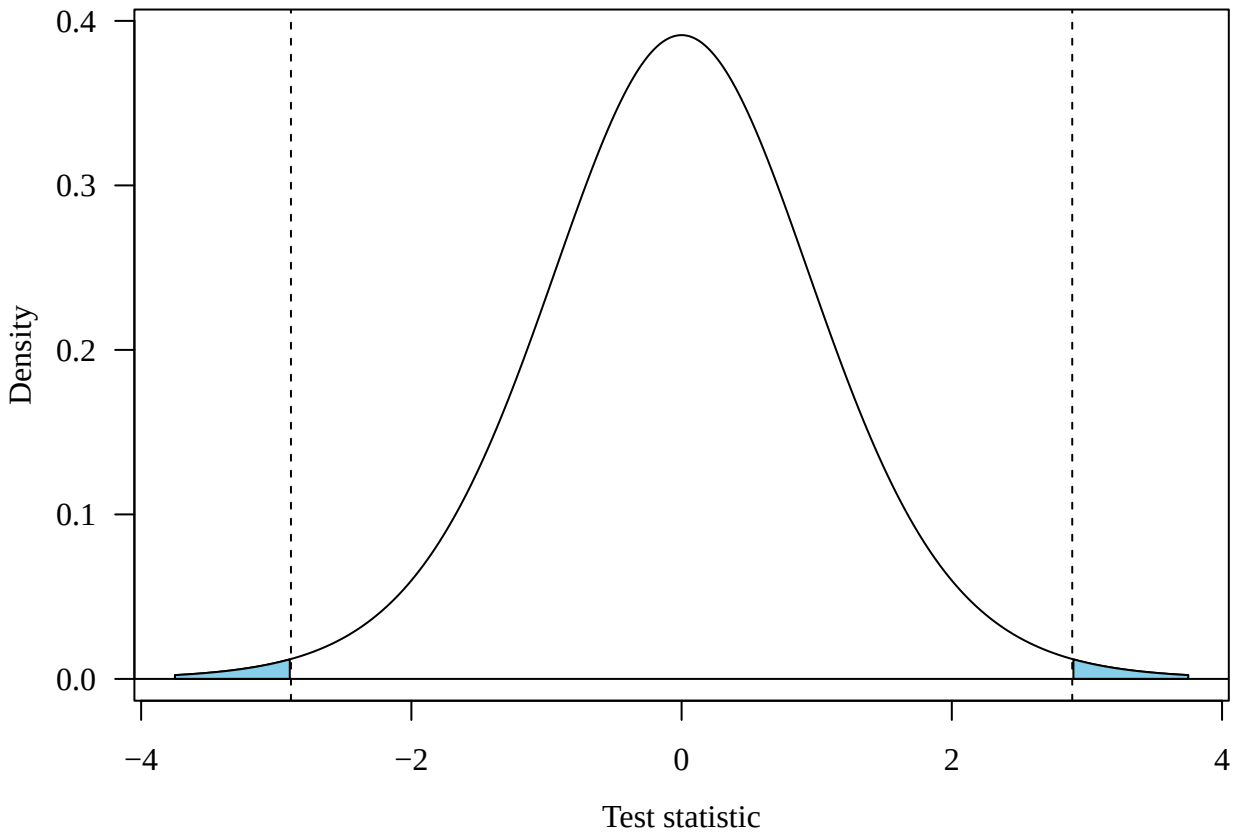
```
# Null hypothesis value
beta1_null <- -1
# Compute test statistic (t*)
# Measures how far estimated beta_1 is from null value in SE units
t_star <- (beta1_hat - beta1_null) / se_beta1
# Compute two-sided p-value using t-distribution with df = n - 2
p_value <- 2 * pt(t_star, n - 2, lower.tail = F)
p_value
```

```
##        Age
## 0.01262031
```

```
par(las = 1, mar = c(3.5, 3.5, 1, 0.5), mgp = c(2.5, 1, 0), family = "serif")

# Create grid for t-distribution
x_grid <- seq(-3.75, 3.75, 0.01)
# Compute density of t-distribution with df = 13
y_grid <- dt(x_grid, 13)
# Plot t-distribution curve
plot(x_grid, y_grid, type = "l", xlab = "Test statistic", ylab = "Density", xlim = c(-3.75, 3.75))
# Shade left tail (extreme values less than -t*)
```

```r
polygon(c(x_grid[x_grid < -t_star], rev(x_grid[x_grid < -t_star])),
        c(y_grid[x_grid < -t_star], rep(0, length(y_grid[x_grid < -t_star]))), col = "skyblue")
# Shade right tail (extreme values greater than t*)
polygon(c(x_grid[x_grid > t_star], rev(x_grid[x_grid > t_star])),
        c(y_grid[x_grid > t_star], rep(0, length(y_grid[x_grid > t_star]))), col = "skyblue")
# Add vertical lines at ±t* (observed test statistic)
abline(v = t_star, lty = 2)
abline(v = -t_star, lty = 2)
# Add horizontal axis reference
abline(h = 0)
```



Additional Guidance for Asynchronous Learning

How to Approach This Lab

- Work through each section sequentially
- Read comments carefully before running code
- Focus on interpretation, not just computation

Reflection Questions

1. What is the goal of this analysis?

2. What patterns do you observe in the results?
3. How would you explain your findings to a non-technical audience?