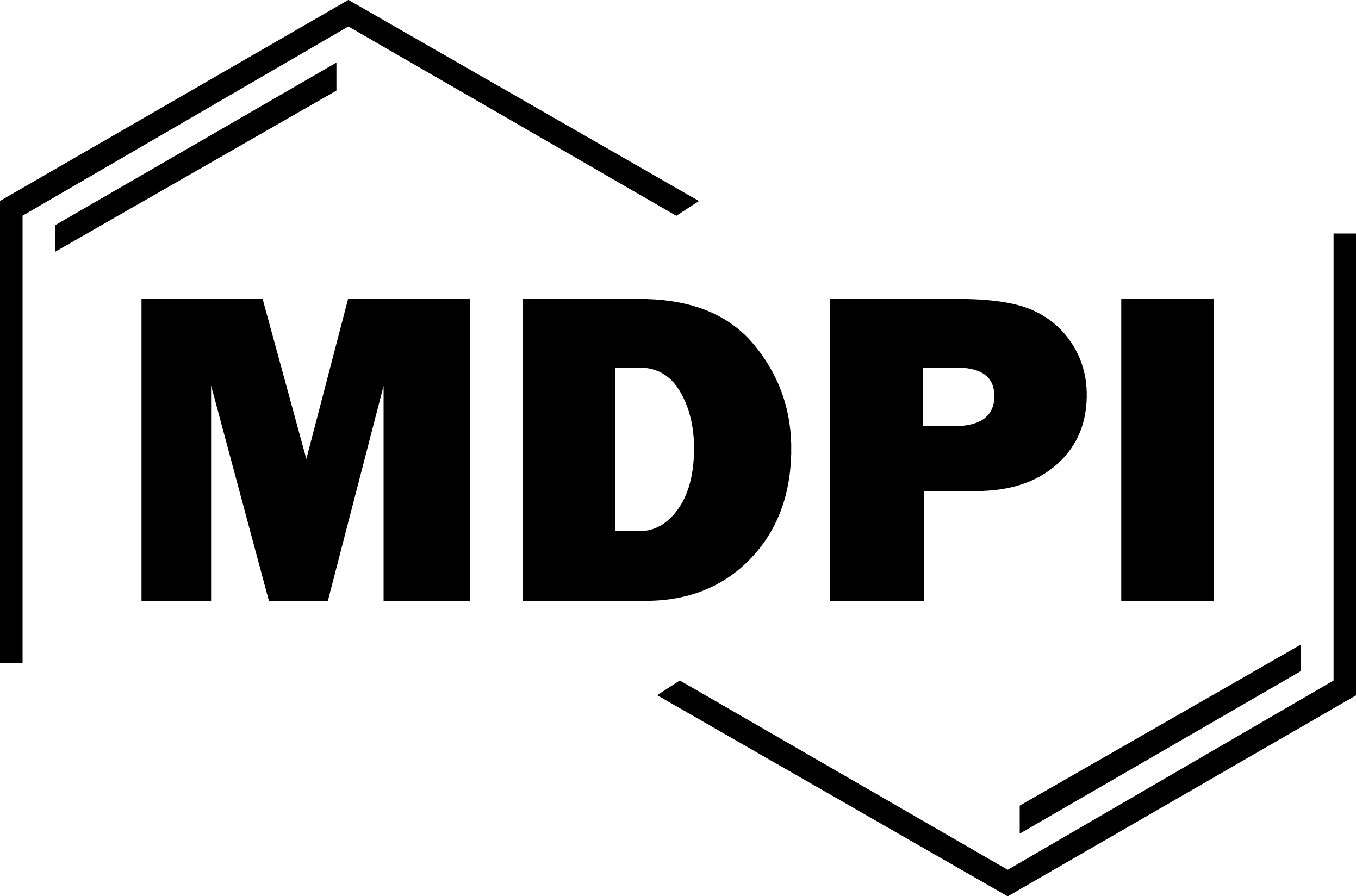
Introduction to Bio-Medical Data Analysis with R

Introduction to Bio-Medical Data Analysis with R

**Biodata with R**

**Nhan Thi Ho**

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

*Author*

Nhan Thi Ho

Vinmec Healthcare System Hanoi, Vietnam

*Editorial Office*

MDPI AG

Grosspeteranlage 5

4052 Basel, Switzerland

For citation purposes, cite as indicated below:

Lastname, A.A.; Lastname, B.B.; Lastname, C.C. *Book Title*. Series Title (optional). Year, Page Range.

**ISBN 978-3-XXXXX-XXX-X (Hbk) ISBN 978-3-XXXXX-XXX-X (PDF)**

[**doi.org/10.3390/books978-3-XXXXX-XXX-X**](https://doi.org/10.3390/books978-3-XXXXX-XXX-X)

© 2025 by the authors. Licensee MDPI, Basel, Switzerland. This book is Open Access and distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

# Contents

[Contents 5](#_Toc218936251)

[List of Figures and Tables 9](#_Toc218936252)

[Abbreviations 13](#_Toc218936253)

[About the Author 15](#_Toc218936254)

[Preface 17](#_Toc218936255)

[Acknowledgements 19](#_Toc218936256)

[1 Introduction 21](#_Toc218936257)

[1.1 About This Book 21](#_Toc218936258)

[1.2 Target audience 21](#_Toc218936259)

[1.3 Main Contents of This Book 22](#_Toc218936260)

[2 Introduction to R 23](#_Toc218936275)

[2.1 What Is R? 23](#_Toc218936276)

[2.2 RStudio 23](#_Toc218936277)

[2.3 Rmarkdown 23](#_Toc218936278)

[2.4 Base R Example 23](#_Toc218936280)

[2.5 Install, Load, Update R Packages 24](#_Toc218936281)

[2.6 Working with R Objects 25](#_Toc218936282)

[2.6.1 Vector 25](#_Toc218936283)

[2.6.2 Matrix 26](#_Toc218936284)

[2.6.3 Array 26](#_Toc218936285)

[2.6.4 List 27](#_Toc218936286)

[2.6.5 Data Frame 27](#_Toc218936287)

[2.6.6 Assessing the Structure of Data Objects 28](#_Toc218936288)

[2.6.7 Useful Functions to Assess R Objects 29](#_Toc218936289)

[2.6.8 Understanding Vectors 29](#_Toc218936290)

[2.7 Managing Biomedical Data in R 30](#_Toc218936291)

[2.7.1 Entering Data 31](#_Toc218936292)

[2.7.2 Importing Data from a File 31](#_Toc218936293)

[2.7.3 Some Other Data Import Functions 32](#_Toc218936294)

[2.7.4 Exporting Data 34](#_Toc218936295)

[2.7.5 Basic Data Manipulation with “tidyverse” 34](#_Toc218936296)

[2.8 Basic Data Visualization 37](#_Toc218936297)

[2.8.1 Barplot 37](#_Toc218936298)

[2.8.2 Histogram 38](#_Toc218936299)

[2.8.3 Density 39](#_Toc218936301)

[2.8.4 Histogram with Density 40](#_Toc218936303)

[2.8.5 Scatter Plot 42](#_Toc218936305)

[2.8.6 Boxplot 43](#_Toc218936307)

[2.8.7 Pie Charts 44](#_Toc218936308)

[2.9 Summary Statistics 45](#_Toc218936309)

[2.9.1 Basic Descriptive Summary Functions 45](#_Toc218936310)

[2.9.2 Basic Statistical Test 47](#_Toc218936311)

[2.9.3 Summary Statistic Table 49](#_Toc218936312)

[2.10 Practice 55](#_Toc218936313)

[2.11 Practice Review 55](#_Toc218936314)

[2.11.1 The Requirements 55](#_Toc218936315)

[2.11.2 Further Exploration 60](#_Toc218936316)

[2.12 Useful Resources 66](#_Toc218936317)

[2.12.1 R Books 66](#_Toc218936318)

[2.12.2 Datasets 66](#_Toc218936319)

[2.13 Chapter Summary 66](#_Toc218936320)

[3 Basic Statistics with R 67](#_Toc218936321)

[3.1 Data Distribution 67](#_Toc218936322)

[3.1.1 The Normal Distribution 67](#_Toc218936323)

[3.1.2 The Binomial Distribution 68](#_Toc218936324)

[3.2 Basic Statistical Tests 70](#_Toc218936325)

[3.2.1 Tests for Comparing Continuous Variables 70](#_Toc218936326)

[3.2.2 Tests for Comparing Categorical Variables 73](#_Toc218936327)

[3.3 Data Summary and Basic Statistical Test in a Table 74](#_Toc218936328)

[3.4 Sample Size—Power Estimation 78](#_Toc218936329)

[3.4.1 Power, Sample Size for Comparing Continuous Variables 78](#_Toc218936330)

[3.4.2 Power, Sample Size for Comparing Categorical Variables 79](#_Toc218936331)

[3.4.3 Sample Size for Estimating Parameter with Desired Precision 80](#_Toc218936332)

[3.5 Chapter Summary 82](#_Toc218936333)

[4 Linear Regression with R 83](#_Toc218936334)

[4.1 Introduction to Linear Regression 83](#_Toc218936335)

[4.2 Simple Linear Regression 83](#_Toc218936336)

[4.2.1 Simple Linear Model with R 83](#_Toc218936337)

[4.2.2 Linear Model Assumptions 87](#_Toc218936338)

[4.2.3 Checking Linear Model Assumptions 87](#_Toc218936339)

[4.2.4 Checking Influential Observations 91](#_Toc218936340)

[4.2.5 Categorical Predictor Variables 92](#_Toc218936341)

[4.3 Multiple Linear Regression 103](#_Toc218936342)

[4.4 Model Selection 109](#_Toc218936343)

[4.5 Model Validation 110](#_Toc218936344)

[4.5.1 Performance Metrics 111](#_Toc218936345)

[4.5.2 Calibration 112](#_Toc218936346)

[4.6 Useful Resources 114](#_Toc218936347)

[4.7 Chapter Summary 114](#_Toc218936348)

[5 Logistic Regression with R 115](#_Toc218936349)

[5.1 Why logistic regression is needed? 115](#_Toc218936350)

[5.2 Logistic Regression Overview 116](#_Toc218936351)

[5.2.1 Statistical Concepts 116](#_Toc218936352)

[5.2.2 Univariable and Multivariable Models 124](#_Toc218936353)

[5.3 Logistic Regression Model Building and Selection 124](#_Toc218936354)

[5.3.1 Main Steps 124](#_Toc218936355)

[5.3.2 Building Model in R 124](#_Toc218936356)

[5.4 Model Evaluation 128](#_Toc218936357)

[5.4.1 Classification Metrics 128](#_Toc218936358)

[5.4.2 Calculate, Visualize Classification Measures 129](#_Toc218936359)

[5.4.3 Discrimination Power 132](#_Toc218936360)

[5.4.4 Model Fit Statistics 133](#_Toc218936361)

[5.5 Model Validation 135](#_Toc218936362)

[5.5.1 Create Training and Test Samples 135](#_Toc218936363)

[5.5.2 Build Logitic Models on Training Data and Predict on Test Data 135](#_Toc218936364)

[5.6 Useful Resources 139](#_Toc218936365)

[5.7 Chapter Summary 139](#_Toc218936366)

[6 Survival Data Analysis with R 141](#_Toc218936367)

[6.1 Introduction to Survival Analysis 141](#_Toc218936368)

[6.1.1 Survival Data 141](#_Toc218936369)

[6.1.2 Key Statistical Concepts 141](#_Toc218936370)

[6.2 Essential R Packages for Survival Analysis 142](#_Toc218936371)

[6.2.1 Working with Dates 142](#_Toc218936372)

[6.2.2 Package “Survival” Main Functions 143](#_Toc218936373)

[6.3 Survival Curves and Models 144](#_Toc218936374)

[6.3.1 Survival Model and Curve for Overall Data 144](#_Toc218936375)

[6.3.2 Comparing Survival Times between Groups 147](#_Toc218936376)

[6.3.3 Hazard ratios 150](#_Toc218936377)

[6.3.4 Multivariate Cox model 151](#_Toc218936378)

[6.3.5 Stratified Cox model 152](#_Toc218936380)

[6.4 Model Evaluation 152](#_Toc218936381)

[6.4.1 Checking Assumptions for Cox Model 152](#_Toc218936382)

[6.4.2 Evaluating Model Accuracy 155](#_Toc218936383)

[6.4.3 Residual Analysis for Goodness-of-Fit 156](#_Toc218936384)

[6.5 Model Selection 158](#_Toc218936385)

[6.5.1 Partial Likelihood Hypothesis Tests 158](#_Toc218936386)

[6.5.2 Information Criteria 158](#_Toc218936387)

[6.5.3 Model Selection Approaches 158](#_Toc218936388)

[6.6 Transform Survival Curves 161](#_Toc218936389)

[6.7 More Advanced Aspects in Survival Analysis 162](#_Toc218936390)

[6.7.1 Competing Risk Analysis 162](#_Toc218936391)

[6.7.2 Time Dependent Covariates 165](#_Toc218936392)

[6.8 Useful Resources 167](#_Toc218936393)

[6.9 Chapter summary 167](#_Toc218936394)

[7 Longitudinal Data Analysis with R 169](#_Toc218936395)

[7.1 Longitudinal Data 169](#_Toc218936396)

[7.2 Example: The Bangladesh Data 169](#_Toc218936397)

[7.2.1 Exploratory Data Analysis 170](#_Toc218936398)

[7.3 Options for Analysis of Change 172](#_Toc218936399)

[7.3.1 Pre-Post Analysis 172](#_Toc218936400)

[7.3.2 Linear Model for Time (the “naïve” approach) 172](#_Toc218936401)

[7.3.3 Issues of Linear Model 175](#_Toc218936402)

[7.4 Generalized Estimating Equations (GEE) 175](#_Toc218936403)

[7.4.1 Introduction to GEE 175](#_Toc218936404)

[7.4.2 GEE Example: The Bangladesh Data 176](#_Toc218936405)

[7.4.3 GEE Example: Other Types of Outcomes 181](#_Toc218936406)

[7.4.4 GEE Summary Main Characteristics 184](#_Toc218936407)

[7.5 Generalized Linear Mixed-Effects Models (GLMM) 184](#_Toc218936408)

[7.5.1 GLMM Overview 184](#_Toc218936409)

[7.5.2 Linear Mixed Effect (LME) Model Example 185](#_Toc218936410)

[7.5.3 GLMM Examples 194](#_Toc218936411)

[7.6 GEE or Mixed Effect Model 198](#_Toc218936412)

[7.7 Useful Resources 199](#_Toc218936413)

[7.7.1 Courses 199](#_Toc218936414)

[7.7.2 Online Tutorials 199](#_Toc218936415)

[7.7.3 Books, Other Resources 199](#_Toc218936416)

[7.8 Chapter Summary 199](#_Toc218936417)

[8 Analysis of Large Consortium Data with R 201](#_Toc218936418)

[8.1 About Data 201](#_Toc218936419)

[8.1.1 Load Required Packages and Example Data 201](#_Toc218936420)

[8.1.2 Summary of data 202](#_Toc218936421)

[8.2 Analysis Approaches 178](#_Toc218936422)

[8.2.1 Estimate Incidence of Cancer by Person Years 178](#_Toc218936423)

[8.2.2 Kaplan-Meier Plot of Time to Cancer 180](#_Toc218936424)

[8.2.3 Cox Models 182](#_Toc218936425)

[8.2.4 Subgroup Analysis 184](#_Toc218936426)

[8.2.5 Analysis of Each Study in the Consortium 188](#_Toc218936427)

[8.2.6 Meta-Analysis Across Studies Based on the Results of Each Study 193](#_Toc218936428)

[8.2.7 Sensitivity Analysis (Leave One Study Out Validation) 195](#_Toc218936429)

[8.3 Useful Resources 202](#_Toc218936430)

[8.4 Chapter Summary 202](#_Toc218936431)

[9 Analysis of Child Growth Data with R 203](#_Toc218936432)

[9.1 Data Summary 203](#_Toc218936433)

[9.1.1 Load Required Packages and Example Data 203](#_Toc218936434)

[9.1.2 Data Summary by Study Sites 205](#_Toc218936435)

[9.2 Analysis Approaches 206](#_Toc218936436)

[9.2.1 Compare Median of Anthropometric Measure for Age by Gender 206](#_Toc218936437)

[9.2.2 Temporal Trend 210](#_Toc218936438)

[9.2.3 Comparison with WHO 2007 Growth References 217](#_Toc218936439)

[9.2.4 Calculate Z-Scores and Prevalence with “Anthro” Package 224](#_Toc218936440)

[9.2.5 Calculate Prevalence of Z-Score Categories with “anthro” Package 227](#_Toc218936441)

[9.2.6 Calculate Prevalence of Nutritional Status Manually 227](#_Toc218936442)

[9.2.7 Calculate Z-Scores Manually 230](#_Toc218936443)

[9.2.8 Calculate Prevalence from z-Scores Manually 238](#_Toc218936444)

[9.2.9 WHO LMS Method for Constructing Normalized Growth Curves 243](#_Toc218936445)

[9.3 Useful Resources 246](#_Toc218936446)

[9.4 Chapter Summary 246](#_Toc218936447)

[10 Analysis of Clinical Trial Data with R 247](#_Toc218936448)

[10.1 Why R for Clinical Trial Data Analysis? 247](#_Toc218936449)

[10.2 Sample Size—Power Estimation for Clinical Trial 247](#_Toc218936450)

[10.2.1 Superiority Trials 247](#_Toc218936451)

[10.2.2 Non-Inferiority Trial 250](#_Toc218936452)

[10.3 Randomization 254](#_Toc218936453)

[10.3.1 Simple Randomization 254](#_Toc218936454)

[10.3.2 Blocked Randomization (Permuted Block Randomization) 255](#_Toc218936455)

[10.3.3 Stratified Randomization 257](#_Toc218936456)

[10.4 Examples of Published Clinical Trial Data Analysis Using R 260](#_Toc218936457)

[10.5 Case Study: example analysis of a vaccine clinical trial data 260](#_Toc218936458)

[10.5.1 Load Required Packages and Example Data 260](#_Toc218936459)

[10.5.2 Baseline Summary 261](#_Toc218936460)

[10.5.3 Time-to-Event Outcome Summary 263](#_Toc218936461)

[10.5.4 Vaccine Efficacy Cox Model 267](#_Toc218936462)

[10.5.5. Safety Data Summary 273](#_Toc218936463)

[10.6 Useful Resources 280](#_Toc218936464)

[10.6.1 Books, Courses 280](#_Toc218936465)

[10.6.2 Journal Articles 280](#_Toc218936466)

[10.7 Chapter Summary 280](#_Toc218936467)

[11 Analysis of Microbiome Data with “*metamicrobiomeR”* 281](#_Toc218936468)

[11.1 Statistical Approaches 281](#_Toc218936469)

[11.2 Example Data 282](#_Toc218936470)

[11.3 *“metamicrobiomeR”* Package Summary 283](#_Toc218936471)

[11.4 Package Usage 283](#_Toc218936472)

[11.4.1 Example Differential Analysis: Comparison Between Breastfeeding Statuses 284](#_Toc218936473)

[11.4.2 Illustration of Meta-Analysis 290](#_Toc218936474)

[11.5 Useful Resources 298](#_Toc218936475)

[11.6 Chapter Summary 298](#_Toc218936476)

[References 299](#_Toc218936477)

[Index 303](#_Toc218936478)

# List of Figures and Tables

[**Figure 1**. Barplot example. 38](#_Toc218932142)

[**Figure 2**. Histogram example. 38](#_Toc218932143)

[**Figure 3**. Plot by group example. 39](#_Toc218932144)

[**Figure 4**. Density example. 39](#_Toc218932145)

[**Figure 5**. Density by group example. 40](#_Toc218932146)

[**Figure 6**. Histogram with density example. 40](#_Toc218932147)

[**Figure 7**. Histogram with density example fixed. 41](#_Toc218932148)

[**Figure 8**. Histogram with density by group. 41](#_Toc218932149)

[**Figure 9**. Scatter plot example. 42](#_Toc218932150)

[**Figure 10**. Scatter plot with regression line. 43](#_Toc218932151)

[**Figure 11**. Scatter plot by group. 43](#_Toc218932152)

[**Figure 12**. Boxplot example. 44](#_Toc218932153)

[**Figure 13**. Box plot with dots. 44](#_Toc218932154)

[**Figure 14**. Pie Charts. 45](#_Toc218932155)

[**Figure 15**. Histogram with base R plot. 47](#_Toc218932156)

[**Figure 16**. Histogram with base R plot 2. 48](#_Toc218932157)

[**Figure 17**. Boxplot number of fecal samples. 57](#_Toc218932158)

[**Figure 18**. Violin plot number of fecal samples. 58](#_Toc218932159)

[**Figure 19**. Bar plot number of fecal samples. 59](#_Toc218932160)

[**Figure 20**. Bar plot with 95%CI. 59](#_Toc218932161)

[**Figure 21**. Histogram by cohort. 60](#_Toc218932162)

[**Figure 22**. whz spaghetti plot. 62](#_Toc218932163)

[**Figure 23**. waz spaghetti plot. 62](#_Toc218932164)

[**Figure 24**. waz spaghetti plot with fitted line. 63](#_Toc218932165)

[**Figure 25**. haz spaghetti plot with fitted line. 64](#_Toc218932166)

[**Figure 26**. All-in-one spaghetti plot. 65](#_Toc218932167)

[**Figure 27**. All-in-one spaghetti plot for each index. 66](#_Toc218932168)

[**Figure 28**. Example normal distribution. 68](#_Toc218932169)

[**Figure 29**. Example binomial distribution. 69](#_Toc218932170)

[**Figure 30**. Q-Q Plot. 71](#_Toc218932171)

[**Figure 31**. Intercept, Slope and Residuals Plot. 86](#_Toc218932172)

[**Figure 32**. Residuals plot. 86](#_Toc218932173)

[**Figure 33**. Residuals vs fitted values plot. 87](#_Toc218932174)

[**Figure 34**. Example trend of residuals. 88](#_Toc218932175)

[**Figure 35**. Residual plot: equal variance. 88](#_Toc218932176)

[**Figure 36**. Residual plot: unequal variance. 89](#_Toc218932177)

[**Figure 37**. Histogram normal distribution. 90](#_Toc218932178)

[**Figure 38**. Q-Q plot normal distribution. 91](#_Toc218932179)

[**Figure 39**. Histogram of continuous variable. 95](#_Toc218932180)

[**Figure 40**. Histogram of continuous variable. 95](#_Toc218932181)

[**Figure 41**. Histogram of continuous variable. 96](#_Toc218932182)

[**Figure 42**. Histogram of continuous variable. 96](#_Toc218932183)

[**Figure 43**. Predicted values by smoker group. 98](#_Toc218932184)

[**Figure 44**. Predicted values by region. 100](#_Toc218932185)

[**Figure 45**. Residual vs. fitted plot 2. 100](#_Toc218932186)

[**Figure 46**. Q-Q plot 2. 101](#_Toc218932187)

[**Figure 47**. Residual vs. fitted plot: log tranformation. 102](#_Toc218932188)

[**Figure 48**. Q-Q plot: log transformation. 103](#_Toc218932189)

[**Figure 49**. Observed and predicted values with 95%CI 106](#_Toc218932190)

[**Figure 50**. Observed and predicted values with interaction. 108](#_Toc218932191)

[**Figure 51**. Calibration plot. 113](#_Toc218932192)

[**Figure 52**. Model outcome vs BMI by linear regression. 116](#_Toc218932193)

[**Figure 53**. Logistic Regression curve. 117](#_Toc218932194)

[**Figure 54**. Predicted Probability of outcome vs. BMI. 119](#_Toc218932195)

[**Figure 55**. Sigmoid curves of univariable vs. multivariable models. 122](#_Toc218932196)

[**Figure 56**. Confusion matrix. 129](#_Toc218932197)

[**Figure 57**. Sensitivity by cutoff. 130](#_Toc218932198)

[**Figure 58**. Specificity by cutoff. 130](#_Toc218932199)

[**Figure 59**. Sensitivity and specificity. 131](#_Toc218932200)

[**Figure 60**. Accuracy by cutoff. 132](#_Toc218932201)

[**Figure 61**. Receiver Operating Characteristics Curve (ROC). 133](#_Toc218932202)

[**Figure 62**. ROC test data. 137](#_Toc218932203)

[**Figure 63**. Sensitivity by cutoff test data. 137](#_Toc218932204)

[**Figure 64**. Specificity by cutoff test data. 138](#_Toc218932205)

[**Figure 65**. Sensitivity, specificity test data. 138](#_Toc218932206)

[**Figure 66**. Accuracy by cutoff test data. 139](#_Toc218932207)

[**Figure 67**. Kaplan-Meier plot using base R. 146](#_Toc218932208)

[**Figure 68**. Kaplan-Meier plot using “survminer” package. 147](#_Toc218932209)

[**Figure 69**. Kaplan-Meier curve between groups with “survminer”. 149](#_Toc218932210)

[**Figure 70**. Check Proportional hazards. 153](#_Toc218932211)

[**Figure 71**. Check linearity. 155](#_Toc218932212)

[**Figure 72**. Martingale Residuals vs. Age. 157](#_Toc218932213)

[**Figure 73**. Deviance Residuals. 157](#_Toc218932214)

[**Figure 74**. Cumulative Incidence Function. 161](#_Toc218932215)

[**Figure 75**. Cumulative hazard function. 162](#_Toc218932216)

[**Figure 76**. Cumulative Incidence of Cancer Progression and Death by Sex. 164](#_Toc218932217)

[**Figure 77**. waz spaghetti plot. 171](#_Toc218932218)

[**Figure 78**. waz mean plot. 172](#_Toc218932219)

[**Figure 79**. waz linear regression plot. 175](#_Toc218932220)

[**Figure 80**. Coefficient estimates. 191](#_Toc218932221)

[**Figure 81**. Random effects. 191](#_Toc218932222)

[**Figure 82**. Predicted values. 192](#_Toc218932223)

[**Figure 83**. Predicted values. 192](#_Toc218932224)

[**Figure 84**. Interaction 193](#_Toc218932225)

[**Figure 85**. Slope. 193](#_Toc218932226)

[**Figure 86**. Residuals. 194](#_Toc218932227)

[**Figure 87**. Kaplan-Meier curve overall pooled data. 182](#_Toc218932228)

[**Figure 88**. Kaplan-Meier curve for each of all studies. 189](#_Toc218932229)

[**Figure 89**. Metaanalysis univariable Cox model. 194](#_Toc218932230)

[**Figure 90**. Metaanalysis multivariable Cox model. 195](#_Toc218932231)

[**Figure 91**. Study left out. 196](#_Toc218932232)

[**Figure 92**. Study left out. 197](#_Toc218932233)

[**Figure 93**. Study left out. 197](#_Toc218932234)

[**Figure 94**. Study left out. 198](#_Toc218932235)

[**Figure 95**. Study left out. 198](#_Toc218932236)

[**Figure 96**. Study left out. 199](#_Toc218932237)

[**Figure 97**. Study left out. 199](#_Toc218932238)

[**Figure 98**. Study left out. 200](#_Toc218932239)

[**Figure 99**. Study left out. 200](#_Toc218932240)

[**Figure 100**. Sensitivity analysis summary: univariable. 201](#_Toc218932241)

[**Figure 101**. BMI median, 95%CI between study sites. 207](#_Toc218932242)

[**Figure 102**. GEE slopes and 95%CI. 215](#_Toc218932243)

[**Figure 103**. Trend slopes, 95%CI. 216](#_Toc218932244)

[**Figure 104**. Trend slopes, 95%CI combined. 217](#_Toc218932245)

[**Figure 105**. Study data height vs. WHO references. 220](#_Toc218932246)

[**Figure 106**. Individual height vs. WHO references. 222](#_Toc218932247)

[**Figure 107**. Centile curves height for age. 244](#_Toc218932248)

[**Figure 108**. Fan-chart height for age. 245](#_Toc218932249)

[**Figure 109**. Cumulative incidence plot. 271](#_Toc218932250)

[**Figure 110**. CIF stratified by risk category. 273](#_Toc218932251)

[**Figure 111**. Taxa mean plot phylum. 285](#_Toc218932252)

[**Figure 112**. Taxa mean plot family. 285](#_Toc218932253)

[**Figure 113**. Metaanalysis nice plot. 294](#_Toc218932254)

[**Figure 114**. Metaanalysis forest plot. 297](#_Toc218932255)

[**Table 1**. Functions to assess R objects. 29](#_Toc218932256)

[**Table 2**. Operations on single vectors. 30](#_Toc218932257)

[**Table 3**. Summary table. 51](#_Toc218932258)

[**Table 4**. More comprehensive summary table. 53](#_Toc218932259)

[**Table 5**. Summary, statistical tests for all variables. 75](#_Toc218932260)

[**Table 6**. Summary, statistical tests, adjusted options. 77](#_Toc218932261)

[**Table 7**. Data summary table. 93](#_Toc218932262)

[**Table 8**. Data summary. 125](#_Toc218932263)

[**Table 9**. Cox model with interaction term. 167](#_Toc218932264)

[**Table 10**. Summary of data by study and overall. 204](#_Toc218932265)

[**Table 11**. Summary of data by main exposure. 177](#_Toc218932266)

[**Table 12**. Incidence by study. 179](#_Toc218932267)

[**Table 13**. Incidence by main exposure status by study. 180](#_Toc218932268)

[**Table 14**. Summary of unadjusted Cox model. 183](#_Toc218932269)

[**Table 15**. Summary of adjusted Cox model. 184](#_Toc218932270)

[**Table 16**. Summary of subgroup analysis results. 190](#_Toc218932271)

[**Table 17**. Univariable Cox model results of each study. 190](#_Toc218932272)

[**Table 18**. Multivariable Cox model results of each study. 192](#_Toc218932273)

[**Table 19**. Data summary by study site. 205](#_Toc218932274)

[**Table 20**. Height median, 95%CI of males. 208](#_Toc218932275)

[**Table 21**. Height median difference, 95%CI of males. 209](#_Toc218932276)

[**Table 22**. Slopes raw table. 213](#_Toc218932277)

[**Table 23**. GEE slopes and 95%CI. 214](#_Toc218932278)

[**Table 24**. Height classification prevalence and 95%CI. 223](#_Toc218932279)

[**Table 25**. Prevalence, 95%CI of stunting. 224](#_Toc218932280)

[**Table 26**. Prevalence of wasting, obesity based on z-scores. 229](#_Toc218932281)

[**Table 27**. Prevalence of stunting based on z-scores. 229](#_Toc218932282)

[**Table 28**. Prevalence, 95%CI Obesity, Wasting calculated manually. 240](#_Toc218932283)

[**Table 29**. Prevalence, 95%CI stunting under 5 calculated manually. 241](#_Toc218932284)

[**Table 30**. Prevalence, 95%CI Obesity, Thinness calculated manually. 242](#_Toc218932285)

[**Table 31**. Prevalence, 95%CI of stunting calculated manually. 242](#_Toc218932286)

[**Table 32**. Summary of baseline data. 263](#_Toc218932287)

[**Table 33**. Event overall. 265](#_Toc218932288)

[**Table 34**. Event by risk. 266](#_Toc218932289)

[**Table 35**. Summary of vaccine efficacy. 269](#_Toc218932290)

[**Table 36**. Summary of serious adverse events. 276](#_Toc218932291)

[**Table 37**. Summary of adverse events by SOC and PT. 279](#_Toc218932292)

[**Table 38**. Functions in the “metamicrobiomeR” package. 283](#_Toc218932293)

# Abbreviations

ACC Accuracy

AE Adverse Event

AIC Akaike Information Criterion

API Application Programming Interface AUC Area Under the Curve

BCa Bias-corrected and accelerated

BEZI zero inflated beta

BIC Bayesian Information Criterion

BMI Body Mass Index

CI Confidence Interval

CHF Cumulative Hazard Function

CIF Cumulative Incidence Function COVID-19 Coronavirus disease 2019

FDR False Discovery Rate

FN False Negative

FP False Positive

GAMLSS Generalized Additive Model for Location, Scale and Shape GEE Generalized Estimating Equation

GLM Generalized Linear Models

GLMM Generalized linear mixed-effects model HR Hazard Ratio

ITT Intention to treat

IQR Interquartile range

IRR Incident Rate Ratios

KEGG Kyoto Encyclopedia of Genes and Genomes KM Kaplan- Meier

LASSO Least Absolute Shrinkage and Selection Operator LME Linear mixed-effects models

MAE Mean Absolute Error

MD Medical Doctor

mITT modified intention to treat

MLE Maximum Likelihood Estimation MSE Mean Squared Error

NPV Negative Predictive Value

OR Odds Ratio

RD Risk Difference

RMSE Root Mean Squared Error

ROC Receiver-operating characteristic curve RR Rate (or Risk) Ratio

PhD Doctor of Philosophy

PFS Progression-Free Survival

PPV Positive predictive value

PT Preferred Term

QIC Quasi Information Criterion

SAE Serious Adverse Event

SBP Systolic blood pressure

SD Standard Deviation

SE Standard Error

SHR Subdistribution Hazard Ratio

SOC System Organ Class

TN True Negative

TP True Positive

URL Uniform Resource Locator

USA United States of America WHO World Health Organization

# About the Author

Nhan Thi Ho, MD, PhD got her PhD degree in (quantitative) Epidemiology at Michigan State University, US. Then, she was a Researcher & Biostatistician at Oxford University Clinical Research Unit, Visiting Scholar in mathematical modeling at Princeton University, US and Fellow in Epidemiology at Columbia University Medical Center (CUMC), US before joining Vinmec Healthcare System in Vietnam as Head of Data Science & Biomedical Research Methodology. She is experienced in Biostatistics for various types of data and is experienced with various statistical software such as R, SAS, and Stata. She is the author and creator of the R package “*metamicrobiomeR*” for analysis and meta-analysis of microbiome data. She is the primary author and editor of a published book entitled “Clinical Research Methodology - Basic”. Alongside with her research position, she is also a Pediatrician with working experience in major children hospitals in Vietnam and France.

# Preface

This book includes a series of hands-on practical introductory guides to analysis of bio-medical data using R. It is designated mainly for bio-medical researchers and students. It will provide readers with basic statistical programming skill in R together with practical data analysis skill and applied statistical knowledge not only for common biostatistical topics but also for some specialized biomedical-clinical research topics and data types. The knowledge and skills from this series are also useful and applicable for those who use other statistical software.

**Nhan Thi Ho**

*Author*

# Acknowledgements

The author would like to thank Vinmec Healthcare System for their support during the organization of the course “Bio-Medical Data Analysis with R: Intro Series” and during the preparation and publication of this book.

# Introduction

## About This Book

Data analysis is an essential part of bio-medical research. To turn research data into meaningful scientific insight and report, researchers need a powerful and flexible analytical tool. R is a free, open-source statistical analysis software that has been extensively used across many fields of bio-medical research. It is well-known not only because it is free, but also because of its power, its graphics capabilities, and its vast ecosystem of community-built packages that provide specialized functions for many statistical challenges. For a bio-medical researcher, being able to use R is a "handy skill" to take control of one's data, from cleaning and visualization to complex modeling and reporting. This book is designed to be a hands-on practical guide to using R for the specific types of data and analytical questions that bio-medical researchers face every day.

This book's practical, "learning-by-doing" approach stems from its origin. During COVID-19 restriction in 2020, Nhan Thi Ho MD, PhD developed and delivered an online course organized by Vinmec Healthcare System: “Bio-Medical Data Analysis with R: Intro Series”. The course's hands-on, practical focus attracted hundreds of participants from Vietnam and other countries, and it was highly evaluated by its participants. This book is a substantive editing of the lectures of the aforementioned course. This book aims to sustain and deliver the course values to a much larger global audience. Furthermore, the novel contribution of this book, which differentiate it from other books about R, are the four chapters on four special topics regarding the analysis for various types of bio-medical and clinical data and study designs which the author Nhan Thi Ho has done and published. The philosophy of this book is not to teach theoretical statistics, but to provide readers with the basic statistical programming skills in R alongside the practical data analysis skills they need to conduct and report their own research.

## Target audience

This book is designated primarily for bio-medical and clinical students and researchers. This may include biologists, physicians and clinicians involved in research, epidemiologists and public health professionals, undergraduate and graduate students in health sciences, biostatistics, or epidemiology. It is also aimed at students and researchers who may be using other statistical software (like SPSS, Stata, or SAS) and wish to transition to R, or who are new to programming and need a practical, applied introduction.

The book is designed as an "Intro Series". It assumes the audience have a basic understanding of core statistical concepts but requires no prior R programming experience. It builds R skills from the ground up, starting with R basics and moving progressively into more advanced, real-world biomedical and clinical data topics. The knowledge and skills provided are also useful and applicable for those who may use other statistical software.

## Main Contents of This Book

The book is structured to build skills progressively, starting with the fundamentals of R before moving into using R for core statistical methods and more advanced, specialized topics that bio-medical researchers encounter.

**Part 1: The Foundation for R**: The book begins with an introduction to the R environment, RStudio, and R Markdown, followed by the basics of R programming, data manipulation, data description and visualization (Chapter 2).

**Part 2: Using R for Core Statistical Methods**: This section covers using R for fundamental statistical concepts, distributions, basic statistical tests, sample size – power estimation (Chapter 3) and for the most common and essential statistical models in bio-medical research: Linear Regression (Chapter 4), Logistic Regression (Chapter 5), Survival (Time-to-Event) Analysis (Chapter 6), and Longitudinal Data Analysis (Chapter 7).

**Part 3: More Advanced Bio-Medical Research Topics**: This section covers using R for Analysis and Meta-analysis of Large Consortium Data (Chapter 8), Analysis of Child Growth Data (Chapter 9), Analysis of Clinical Trial Data (Chapter 10), Analysis and Meta-analysis of Microbiome Data with “*metamicrobiomeR”* package (Chapter 11).

After reading and following the examples in this book, readers will gain the applied statistical knowledge and practical skills necessary to analyze and report their own bio-medical and clinical research data using R.



# Introduction to R

This chapter introduces bio-medical researchers to R and R packages. It covers basic data handling, description, summary and visualization. It also includes a section for practice exercises and practice review to help readers acquire the skills smoothly. Readers do not need prior programming skill in R or other software to start learning.

## What Is R?

R is a free software environment for statistical computing and graphics. R is open source and thus is continuously being improved by a large community of developers. It has a huge ecosystem of community-built packages that help tackle various statistical matters. To download, install, and learn more, go to [R cran](https://www.r-project.org/) [1].

## RStudio

RStudio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management. To download, install, and learn more, go to [RStudio page](https://rstudio.com/products/rstudio/download/) [2].

## Rmarkdown

R Markdown is a file format for making dynamic documents with R.

It turns your analyses into high quality REPRODUCIBLE documents, reports, presentations and dashboards. [3].

## Base R Example

R comes with a set of base packages that provide fundamental functions. Here are some examples of how to use R for basic calculations, data generation, and summary statistics.

You can use R as a calculator to perform simple arithmetic operations directly in the R console.

*# Use R as a calculator:*

((2+2)\*3)/2

## [1] 6

R has a vast number of built-in functions. For example, the *sqrt()* function calculates the square root of a number.

sqrt(9) ## [1] 3

mean(c(2,4,5,9)) ## [1] 5

We can check how to use a function by adding “?” in front of a function. For example, use the function *rnorm()* to generate data, *summary()* to summarize the data and *t.test()* to perform a t-test on the data.

*#Check how to use a function #?rnorm*

*#use the function*

data1<- rnorm(n = 30,mean = 1.5, sd = 0.5)

summary(data1)

## Min. 1st Qu. Median Mean 3rd Qu. Max. ## 0.6094 1.1069 1.3362 1.4283 1.7080 2.3906

data2<-rnorm(n = 35, mean = 1.8,sd = 1)

summary(data2)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## -1.335 1.060 1.742 1.700 2.432 3.383

Basic statistical test may be done easily in R, for example we can do a t-test:

*#?t.test*

t.test(x = data1, y = data2, alternative = "two.sided", var.equal = FALSE, conf.level = 0.95)

##

## Welch Two Sample t-test

##

## data: data1 and data2

## t = -1.3973, df = 48.737, p-value = 0.1687

## alternative hypothesis: true difference in means is not

## equal to 0

## 95 percent confidence interval:

## -0.6633834 0.1192729

## sample estimates:

## mean of x mean of y

## 1.428314 1.700369

## Install, Load, Update R Packages

R's functionality can be extended by installing R packages.

* In RStudio: You can install packages by navigating to Packages > Install and then typing the package name.
* From R console: You can use the *install.packages()* function by typing: *install.packages("packagename")*.

The example below shows how to install and load a single package which is my R package “*metamicrobiomeR*” [4]:

*#install.packages("metamicrobiomeR")*

*#load package for use*

library("metamicrobiomeR")

The example below shows how to install and load multiple packages. This code creates a vector of package names *("knitr" [5], "rmarkdown" [6], "installr” [7], “haven" [8], "readr" [9], "yaml” [10], tidyverse" [11], "kableExtra” [12]*) and then uses *lapply()* to load each of them:

*#install multiple packages*

Packages <- c("knitr","rmarkdown","installr","haven", "readr","yaml","tidyverse", "kableExtra")

*#install.packages(Packages)*

*#load multiple packages*

lapply(Packages, library, character.only = TRUE)

The below commands are useful for updating packages:

*# download and install newest version of the package #update.packages()*

*# shows which packages have updates #old.packages ()*

*# looks for new packages that are not already installed*

*#new.packages ()*

The below commands may be used to get help for package, function, data:

*#help for package*

*#help(package = "metamicrobiomeR")*

*#help for function*

*#help("taxa.compare", package = "metamicrobiomeR")*

*#or load package*

*#library(metamicrobiomeR) #?taxa.compare*

*#help for data*

*#help(tabsex4, package = "metamicrobiomeR")*

*# display data help page*

## Working with R Objects

R has several fundamental data structures, also known as objects.

### Vector

A vector is a collection (one-dimensional array) of elements of the same data type:

*# A vector of numeric elements*

x<-c(1,3,4,7,8,10)

x

## [1] 1 3 4 7 8 10

*# A vector of character elements*

a<-c("red","blue","white")

a

## [1] "red" "blue" "white"

*#creating vector*

seq(1,10, by = 2)

## [1] 1 3 5 7 9

rep("boring",3)

## [1] "boring" "boring" "boring"

### Matrix

A matrix is a 2-dimensional representation of a vector (two-dimensional rectangular data set) where all elements are of the same data type.

The example code below reshapes the vector *x* above into a matrix with 2 rows:

|  |  |  |  |
| --- | --- | --- | --- |
| y<-matrix(x, y | | nrow | = 2) |
| ## | [,1] | [,2] | [,3] |
| ## | [1,] 1 | 4 | 8 |
| ## | [2,] 3 | 7 | 10 |

### Array

An array, similar to a matrix, is an n dimensional representation of a vector. This code creates a 3-dimensional array from the vector *x*:

z <- array(x, dim = c(1, 3, 2)); z

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ##  ## ## | , , 1 | [,1] | [,2] | [,3] |
| ## | [1,] | 1 | 3 | 4 |
| ## |  |  |  |  |
| ## | , , 2 |  |  |  |
| ## |  |  |  |  |
| ## |  | [,1] | [,2] | [,3] |
| ## | [1,] | 7 | 8 | 10 |
|  |  |  |  |  |

### List

A list is a collection of “bins”, a versatile object that can contain any kind of R object of any data types and structures. This code creates a list containing a vector, a matrix, an array, and another vector:

mylist <- list(x, y, z, a); mylist

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | [[1]] | |  | |
| ## | [1] 1 3 | | 4 7 8 10 | |
| ## |  | |  | |
| ## | [[2]] | |  | |
| ## | [,1] | | [,2] [,3] | |
| ## | [1,] | 1 | 4 | 8 |
| ## | [2,] | 3 | 7 | 10 |
| ## |  |  |  |  |
| ## | [[3]] |  |  |  |
| ## | , , 1 |  |  |  |
| ## |  |  |  |  |
| ## |  | [,1] | [,2] | [,3] |
| ## | [1,] | 1 | 3 | 4 |
| ## |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| ##  ## ## | , , 2  [,1] | [,2] [,3] |
| ## | [1,] 7 | 8 10 |
| ## |  |  |
| ## |  |  |
| ## | [[4]] |  |
| ## | [1] "red" | "blue" "white" |
|  |  |  |

### Data Frame

A data frame is a list in tabular form where each “bin” contains a data vector of the same length. A data frame is the usual tabular data set familiar to epidemiologists. Each row is a record and each column (“bin”) is a field. Both matrix and data frame are tabular data-types.

* + - * All columns in a matrix must have the same data type (numeric, character, etc.) and the same length.
      * A data frame is more general than a matrix, in that different columns can have different modes (numeric, character, factor, etc.). Just like a table in a database or excel sheet.

This code creates a data frame with three columns: smoker, cancer, and quant, and then displays the first four rows.

smoker <- c("Y", "Y", "N", "N", "Y", "Y", "Y", "N", "Y")

cancer <- c("Y", "N", "N", "Y", "Y", "Y", "N", "N", "Y") quant<-c(1,2,3,4,5,6,7,8,9)

mydf <- data.frame(smoker, cancer,quant)

mydf[1:4,] *# display rows 1 to 4*

|  |  |  |  |
| --- | --- | --- | --- |
| ## | smoker | cancer | quant |
| ## 1 | Y | Y | 1 |
| ## 2 | Y | N | 2 |
| ## 3 | N | N | 3 |
| ## 4 | N | Y | 4 |
|  |  |  |  |

### Assessing the Structure of Data Objects

Understanding the structure of an R object is crucial for effective data manipulation and analysis. The *str()* function provides a concise summary of the internal structure of any R object, including its class, dimensions, and the data type of its contents. The output helps quickly identify the variables and their types, which is essential before performing any operation (e.g., ensuring numeric variables are actually stored as num).

data("tabsex4", package = "metamicrobiomeR")

str(tabsex4)

## 'data.frame': 701 obs. of 23 variables:

## $ id : chr "kbacteria.pactinobacteria" "kbacteria.pbacteroidetes" "kbacteria.pfirmicutes" "kbacteria.pfusobacteria" ...

## $ Estimate.genderMale : num 0.2269 -0.1549 -0.056 -0.0648 -0.0887 ...

## $ Std. Error.genderMale : num 0.121 0.119 0.104 0.282 0.118 ...

## $ t value.genderMale : num 1.874 -1.305 -0.537 -0.23 -0.751 ...

## $ Pr(>|t|).genderMale : num 0.0619 0.1931 0.5919 0.8186 0.4532 ...

## $ Estimate.bfNon\_exclusiveBF : num -0.3688 0.2557 0.2362 -0.0605 0.3699 ...

## $ Std. Error.bfNon\_exclusiveBF : num 0.139 0.136 0.116 0.364 0.134 ...

|  |  |  |  |
| --- | --- | --- | --- |
| ## | $ t value.bfNon\_exclusiveBF | : num | -2.647 1.887 2.035 -0.166 2.764 ... |
| ## | $ Pr(>|t|).bfNon\_exclusiveBF | : num | 0.00857 0.0602 0.04276 0.86812 0.00606 ... |
| ## | $ Estimate.bfNo\_BF | : num | -1.329 0.135 0.356 -0.403 1.153 ... |
| ## | $ Std. Error.bfNo\_BF | : num | 0.412 0.499 0.392 1.083 0.407 ... |
| ## | $ t value.bfNo\_BF | : num | -3.223 0.27 0.908 -0.372 2.834 ... |
| ## | $ Pr(>|t|).bfNo\_BF | : num | 0.00141 0.7873 0.36478 0.7103 0.0049 ... |
| ## | $ Estimate.age.sample | : num | 0.1251 0.0242 0.0115 0.0285 -0.1348 ... |
| ## | $ Std. Error.age.sample | : num | 0.0377 0.0367 0.0319 0.0857 0.038 ... |
| ## | $ t value.age.sample | : num | 3.318 0.66 0.359 0.333 -3.549 ... |
| ## | $ Pr(>|t|).age.sample | : num | 0.001022 0.510068 0.719525 0.73934 0.000448 ... |
| ## | $ pval.adjust.genderMale | : num | 0.48 0.899 0.961 0.961 0.961 ... |

## $ pval.adjust.bfNon\_exclusiveBF: num 0.059 0.2333 0.1767 0.9976 0.0448 ...

## $ pval.adjust.bfNo\_BF : num 0.0219 1 1 1 0.0468 ...

## $ pval.adjust.sample : num 0.00786 0.82595 0.85375 0.85375 0.00505 ... ## $ study : chr "Subramanian et al 2014 (Bangladesh)"

"Subramanian et al 2014 (Bangladesh)" "Subramanian et al 2014 (Bangladesh)" "Subramanian et al 2014 (Bangladesh)" ...

## $ pop : chr "Bangladesh" "Bangladesh" "Bangladesh" "Bangladesh" ...

### Useful Functions to Assess R Objects

R provides a variety of functions to inspect the characteristics of data objects.

**Table 1**. Functions to assess R objects.

**Function Returns**

*str* summary of data object structure

*attributes* list with data object attributes

*mode* mode of object

type of object; similar to mode but includes double and integer, if applicable

*typeof*

*length* length of object

*class* class of object, if it exists

*dim* vector with object dimensions, if applicable

*nrow* number of rows, if applicable

*ncol* number of columns, if applicable

*dimnames* list containing vectors of names for each dimension, if applicable *rownames* vector of row names of a matrix-like object

*colnames* vector of column names of a matrix-like object

*names* vector of names for the list (for a data frame returns field names) *row.names* vector of row names for a data frame

Source: Author's compilation based on data from the R Manuals.

### Understanding Vectors

Operation of vectors: When you perform an operation on two vectors of the same length, the operation is applied element-wise.

*#Generate vector*

x1 <- 5; y1 <- c(10, 20, 30)

x2 <- c(5, 5, 5); y2 <- c(10, 20, 30)

x1 \* y1

## [1] 50 100 150

x2 \* y2

## [1] 50 100 150

Boolean Queries and Subsetting of Vectors: You can use logical operators *(<, >, <=, >=, ==* for equals, *!=* for not equals, *&* for AND, | for OR) to create a logical vector, and then use that vector to select a subset of elements from another vector. You can assign names to the elements of a vector, which can make the output more informative.

lecturer <- c('Nhan', 'Hien', 'Phuong', 'Tien', 'Duy')

lecturer

## [1] "Nhan" "Hien" "Phuong" "Tien" "Duy"

ms.ages <- c(41, 39, 33, 26, 25)

ms.ages

## [1] 41 39 33 26 25

ms.ages > = 30 *# logical vector for lecturers with ages > = 30*

## [1] TRUE TRUE TRUE FALSE FALSE

thirtysomething <- (ms.ages > = 30) & (ms.ages < 40) lecturer[thirtysomething] *# indexing using logical*

## [1] "Hien" "Phuong"

*#naming vector*

names(ms.ages)<-lecturer

ms.ages

## Nhan Hien Phuong Tien Duy ## 41 39 33 26 25

Operations on Single Vectors: Below are some operations on single vectors.

**Table 2**. Operations on single vectors.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Description** | **Function** | **Description** |
| *sum* | summation | rev | reverse order |
| *cumsum* | cumulative sum | order | order |
| *diff* | x[i + 1] − x[i] | sort | sort |
| *prod* | product | rank | rank |
| *cumprod* | cumulative product | sample | random sample |
| *mean* | mean | quantile | percentile |
| *median* | median | var | variance, covariance |
| *min* | minimum | sd | standard deviation |
| *max* | maximum | table | tabulate character vector |
| *range* | range | xtabs | tabulate factor vector |

Source: Author's compilation based on data from the R Manuals.

## Managing Biomedical Data in R

This section focuses on methods for getting your data into R and preparing it for use.

### Entering Data

For small, simple datasets, you can enter data directly into R using functions like *c()* for vectors and *data.frame()* to combine them into a data frame.

subjname <- c('Lop', 'Chung', 'Minh') subjno <- 1:length(subjname)

age <- c(34, 56, 56)

sex <- c('Male', 'Male', 'Female')

dat <- data.frame(subjno, subjname, age, sex); dat

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | subjno | subjname | age | sex |
| ## | 1 1 | Lop | 34 | Male |
| ## | 2 2 | Chung | 56 | Male |
| ## | 3 3 | Minh | 56 | Female |

### Importing Data from a File

The example below imports the dataset of Bangladeshi children [13] from local directory:

*#your path to dataset*

fdir1\_local<-"./metamicrobiomeR-master/inst/extdata/QIIME\_outputs/Bangladesh/"

*#Read in text file*

dat1\_local<-read.delim(paste0(fdir1\_local,

"Subramanian\_et\_al\_mapping\_file.txt"), header = T,sep = "\t")

*fdir1\_local* stores the file path to the directory. *paste0()* concatenates the directory path and the file name to create the full file path string. *read.delim()* is a base R function used to read delimited text files (*header = T* specifies that the first row contains column names; *sep = "\t"* specifies that the data fields are separated by a tab character)

The example below imports the same dataset from Github:

*# your path to dataset*

*# break down the url for readability*

base\_url <- "https://raw.githubusercontent.com/nhanhocu/metamicrobiomeR/master/" data\_path <- "inst/extdata/QIIME\_outputs/Bangladesh/"

fdir1 <- paste0(base\_url, data\_path)

*# Read in text file*

dat1<-read.delim(paste0(fdir1,"Subramanian\_et\_al\_mapping\_file.txt"), header = T,sep = "\t")

The example below imports another dataset from URL:

base\_url <- "https://stats.idre.ucla.edu/wp-content/uploads/2016/02/"

data\_use <- "test-1.csv"

*#Read in csv file*

test.csv <- read.csv(paste0(base\_url, data\_use), header = T)

### Some Other Data Import Functions

The “*rio”* [14] package provides a convenient, single function, *import()*, that can handle various file formats (CSV, Excel, SAS, Stata, etc.), making data import much easier! All you need is *import()*!

*#install.packages("rio")*

*#install.packages(c('csvy', 'feather', 'fst', 'hexView', 'readODS', 'rmatio'))*

library(rio)

test1<-import(paste(fdir1,"Subramanian\_et\_al\_mapping\_file.txt",sep = ""))

We can import one specified excel sheet using *import()* and import the whole excel files with multiple sheets using *import\_list()*:

*#import always get only 1 dataframe*

test<-import(paste(fdir1,"nature13421-s2.xlsx",sep = ""),sheet = 1,skip = 1)

*#Import multiple sheets of an excel file and create a list of dataframes*

list1<-import\_list(paste(fdir1,"nature13421-s2.xlsx",sep = ""))

*#str(list1)*

*#View(list1[[1]])*

We can also import multiple files from a local directory by using *Sys.glob()* to find specified files in a directory and *import\_list()* to import all of them:

*# Get folder path*

*#fdir1\_local<-"./metamicrobiomeR-master/inst/extdata/QIIME\_outputs/Bangladesh/"*

*#Sys.glob() allows you to find files using a regular expression*

dir1\_local<-Sys.glob(paste(fdir1\_local,"alpha\_div\_collated/\*.txt",sep = "")) list2<-import\_list(dir1\_local)

*#str(list2)*

The codes below will import those same files from github using GitHub API. First, we construct the GitHub API URL for the directory contents. Then we make a GET request to the API and check if the request was successful. Then we parse the JSON response into a data frame and filter the data frame to find files matching the pattern. Then we extract the direct download URLs *'final\_urls'* for these files and use this *'final\_urls'* vector to import these files using *import\_list()*. We need to use three R packages for this task along with the “*rio*” package mentioned above: *"httr"* [15]*, "jsonlite"* [16]*, "dplyr"* [17].

*# Install necessary packages if you don't have them*

*# install.packages("httr")*

*# install.packages("jsonlite")*

*# install.packages("dplyr")*

library(httr) library(jsonlite) library(dplyr)

*# Construct the GitHub API URL for the directory contents*

*# The format is:*

*# https://api.github.com/repos/OWNER/REPO\_NAME/contents/PATH\_TO\_FOLDER*

owner <- "nhanhocu"

repo <- "metamicrobiomeR"

path <- "inst/extdata/QIIME\_outputs/Bangladesh/alpha\_div\_collated"

api\_url <- paste0("https://api.github.com/repos/",

owner, "/", repo, "/contents/", path)

*# Make a GET request to the API*

response <- GET(api\_url)

*# Check if the request was successful*

if (http\_status(response)$category ! = "Success") { stop("Failed to retrieve directory contents from GitHub API.")

}

*# Parse the JSON response into a data frame*

*# The content is in the body of the response, which we extract and parse*

file\_list\_df <- fromJSON(content(response, "text"), flatten = TRUE)

*# Filter the data frame to find files matching the pattern "\*.txt"*

*# We also want to ensure we only get files, not subdirectories*

txt\_files\_df <- file\_list\_df %>%

filter(type = = "file", endsWith(name, ".txt"))

*# Extract the direct download URLs for these files*

*# The 'download\_url' column contains the raw file links*

final\_urls <- txt\_files\_df$download\_url

*# Print the resulting list of URLs*

print(final\_urls)

*# You can now use this 'final\_urls' vector to import the files, for example:*

library(rio)

list2\_github <- import\_list(final\_urls)

### Exporting Data

Once your data is clean and prepared, you will often need to save it for later use or sharing.

*#save one dataframe as an csv file*

*#write.csv(dat1,file = paste(fdir1,"dat1.csv",sep = ""), row.names=F)*

*#save multiple data objects*

*#save(dat1,list1,list2, file = paste(fdir1,"manyfile.rda", sep = ""))*

*#load saved files*

*#print(load(paste(fdir1,"manyfile.rda", sep = "")))*

### Basic Data Manipulation with “tidyverse”

“*tidyverse*” includes a collection of open source packages for the R programming language that share similar design philosophy, grammar, and data structures for tidy data [11]. The key packages include “*dplyr”* [17] for data manipulation and “*ggplot2”* [18] for visualization.

#### Pipes

The pipe operator (%>%) takes the output of a command, and feed it as the input of another command. This allows you to chain multiple data manipulation steps together in a clear, linear, and readable sequence.

library(tidyverse)

*#pipe from library(magrittr), loaded along with other packages*

9 %>% sqrt %>% `+`(8)

## [1] 11

#### Subset Data

Subset data by column and row: The “*dplyr”* package provides intuitive functions for filtering rows with *filter()* and selecting columns with *select()*:

dat2a<-test %>%

slice(1:50) %>%

select(-starts\_with("..."))

We can change column names with *rename\_at()*:

*#change column name*

colnames(dat2a)<-tolower(colnames(dat2a)) oldname<-colnames(dat2a)

newname<-c("child.id",

"family.id",

"cohort",

"gender", "zygosity", "whz","waz","haz",

"age.first.fe",

"age.last.fe",

"no.fe.sam",

"sam.int.mean",

"month.ebf", "age.first.solid", "no.diarhea.yr", "pc.days.diarrhea", "frac.sam.atb.7day", "set")

dat2a<-dat2a %>% rename\_at(vars(oldname), ~newname)

*#colnames(dat2a)*

We can also subset data by row:

dat.tw<-dat2a %>% filter(cohort == "Healthy Twins & Triplets")

*#more filter*

dat.tw<-dat2a %>%

filter(cohort == "Healthy Twins & Triplets")%>% filter(!is.na(zygosity))%>%

filter(between(whz,-1,1))

*#or*

dat.tw<-dat2a %>%

filter(cohort == "Healthy Twins & Triplets" &

!is.na(zygosity) & between(whz,-1,1))

We can also select columns:

*#Base R*

*#Select by position (cautious!)*

subdat<-dat2a[,c(1,4:6)]

*#or select specific column names*

subdat<-dat2a[,c("child.id","gender","zygosity","whz")]

*#pipes and select()*

subdat<-dat2a %>% select(c(1,4:6))

subdat<-dat2a %>% select(child.id,gender,zygosity,whz)

Or discard columns:

*#Base R*

subdat<-dat2a[,-c(1,4:6)]

subdat<-dat2a[,colnames(dat2a)[!colnames(dat2a) %in%

c("child.id", "gender",

"zygosity", "whz")]]

*#or with tidyverse*

subdat<-dat2a %>% select(-c(1,4:6))

subdat<-dat2a %>% select(-c("child.id", "gender",

"zygosity", "whz"))

Or rename columns with *rename()*:

*#Base R*

*#colnames(dat2a)[colnames(dat2a) == "child.id"]<-"childid"*

*#with tidyverse*

dat2a<-dat2a %>%

select(childid = child.id, everything()) dat2a<-dat2a %>%

rename(child.id = childid)

#### Editing data

The *mutate()* function from “*dplyr”* package is used to add new variables or modify existing ones.

*#initial look at data*

summary(dat2a)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | child.id | | family.id cohort | | | |
| ## | Length:50 | | Length:50 Length:50 | | | |
| ## | Class :character | | Class :character Class :character | | | |
| ## | Mode :character | | Mode :character Mode :character | | | |
| ## |  | |  | | | |
| ## |  | |  | | | |
| ## |  | |  | | | |
| ## | gender | | zygosity whz | | | |
| ## | Length:50 | | Length:50 Min. :-1.73842 | | | |
| ## | Class :character | | Class :character 1st Qu.:-0.88598 | | | |
| ## | Mode :character | | Mode :character Median :-0.47063 | | | |
| ## |  | | Mean :-0.42223 | | | |
| ## |  | | 3rd Qu.:-0.03458 | | | |
| ## |  | | Max. : 1.20133 | | | |
| ## | waz | | haz age.first.fe | | | |
| ## | Min. :-3.7555 | | Min. :-4.6586 Min. : 1.00 | | | |
| ## | 1st Qu.:-2.6037 | | 1st Qu.:-3.4314 1st Qu.: 5.00 | | | |
| ## | Median :-1.5116 | | Median :-1.5749 Median : 6.00 | | | |
| ## | Mean :-1.5148 | | Mean :-1.8570 Mean : 9.78 | | | |
| ## | 3rd Qu.:-0.7189 | | 3rd Qu.:-0.7189 3rd Qu.: 9.75 | | | |
| ## | Max. : 1.6698 | | Max. : 1.6698 Max. :37.00 | | | |
| ## | age.last.fe | | no.fe.sam sam.int.mean | | | |
| ## | Min. | :286.0 | Min. | : 9.00 | Min. | :22.56 |
| ## | 1st Qu.:455.0 | | 1st Qu.:18.00 | | 1st Qu.:28.29 | |
| ## | Median :700.5 | | Median :21.00 | | Median :30.48 | |
| ## | Mean :600.6 | | Mean :19.92 | | Mean :31.34 | |
| ## | 3rd Qu.:706.0 | | 3rd Qu.:23.00 | | 3rd Qu.:33.27 | |

## Max. :738.0 Max. :26.00 Max. :42.29

|  |  |  |
| --- | --- | --- |
| ## | month.ebf | age.first.solid no.diarhea.yr |
| ## | Min. :0.0000 | Min. :1.216 Min. :0.000 |
| ## | 1st Qu.:0.1971 | 1st Qu.:4.920 1st Qu.:1.271 |
| ## | Median :1.0348 | Median :6.932 Median :2.916 |
| ## | Mean :2.0716 | Mean :6.690 Mean :2.971 |
| ## | 3rd Qu.:3.2523 | 3rd Qu.:8.361 3rd Qu.:4.054 |
| ## | Max. :7.3260 | Max. :9.790 Max. :8.814 |
| ## | pc.days.diarrhea | frac.sam.atb.7day set |
| ## | Min. : 0.000 | Min. :0.00000 Length:50 |
| ## | 1st Qu.: 1.163 | 1st Qu.:0.08083 Class :character |
| ## | Median : 2.742 | Median :0.11882 Mode :character |
| ## | Mean : 3.125 | Mean :0.18328 |
| ## | 3rd Qu.: 4.314 | 3rd Qu.:0.28373 |
| ## | Max. :10.826 | Max. :0.50000 |

dat2a<-dat2a %>%

mutate(zygosity = replace(zygosity,zygosity == "NA","singleton")) %>%

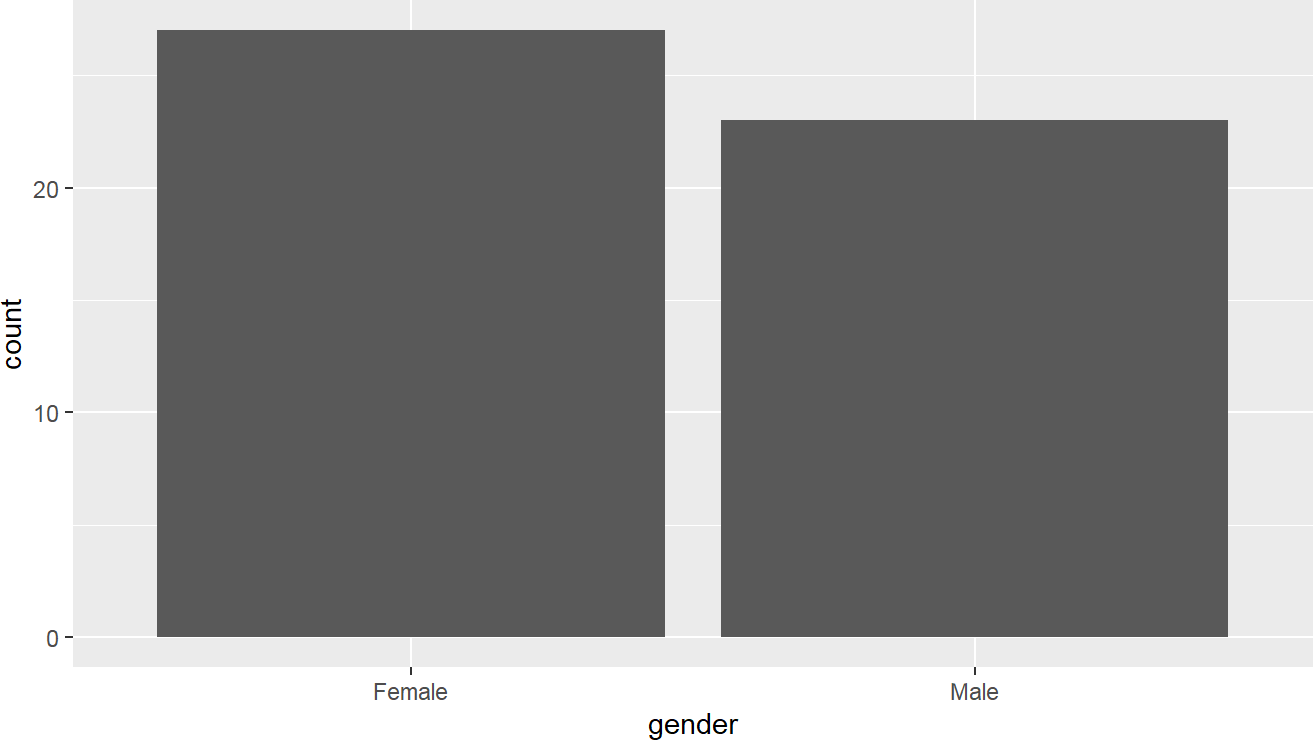
mutate(zygosity = replace(zygosity,zygosity == "not tested",NA))

## Basic Data Visualization

Data visualization is an essential step in biomedical data analysis. The “*ggplot2*” package (part of “*tidyverse*”) is the most popular and powerful tool for creating publication-quality graphics in R. “*ggplot2*” uses a grammar of graphics that builds plots layer by layer [18]. Below are some examples of basic data visualization with “*ggplot2*”.

### Barplot

We can create a barplot with *geom\_bar()*:



library(ggthemes) ggplot(dat2a,aes(gender))+

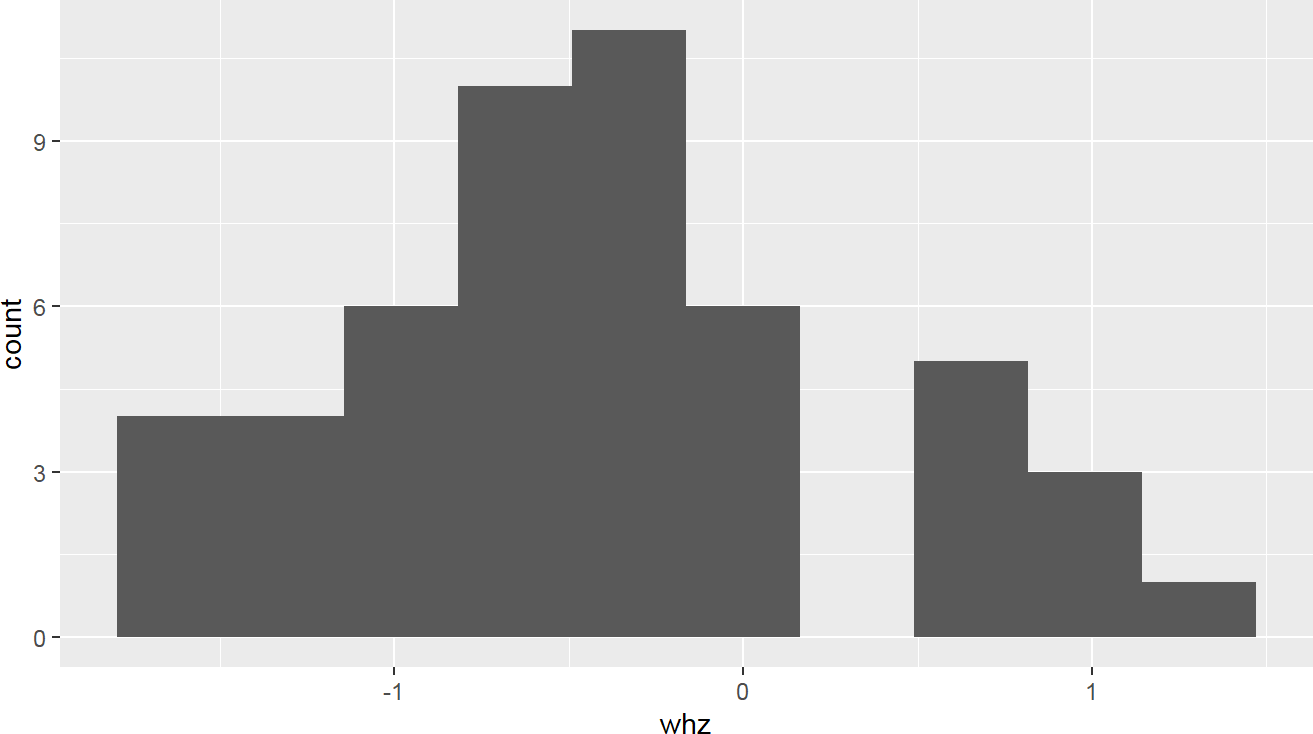
geom\_bar()

Source: Figure by author(s).

**Figure 1**. Barplot example.

### Histogram

We can create a histogram with *geom\_histogram()*:

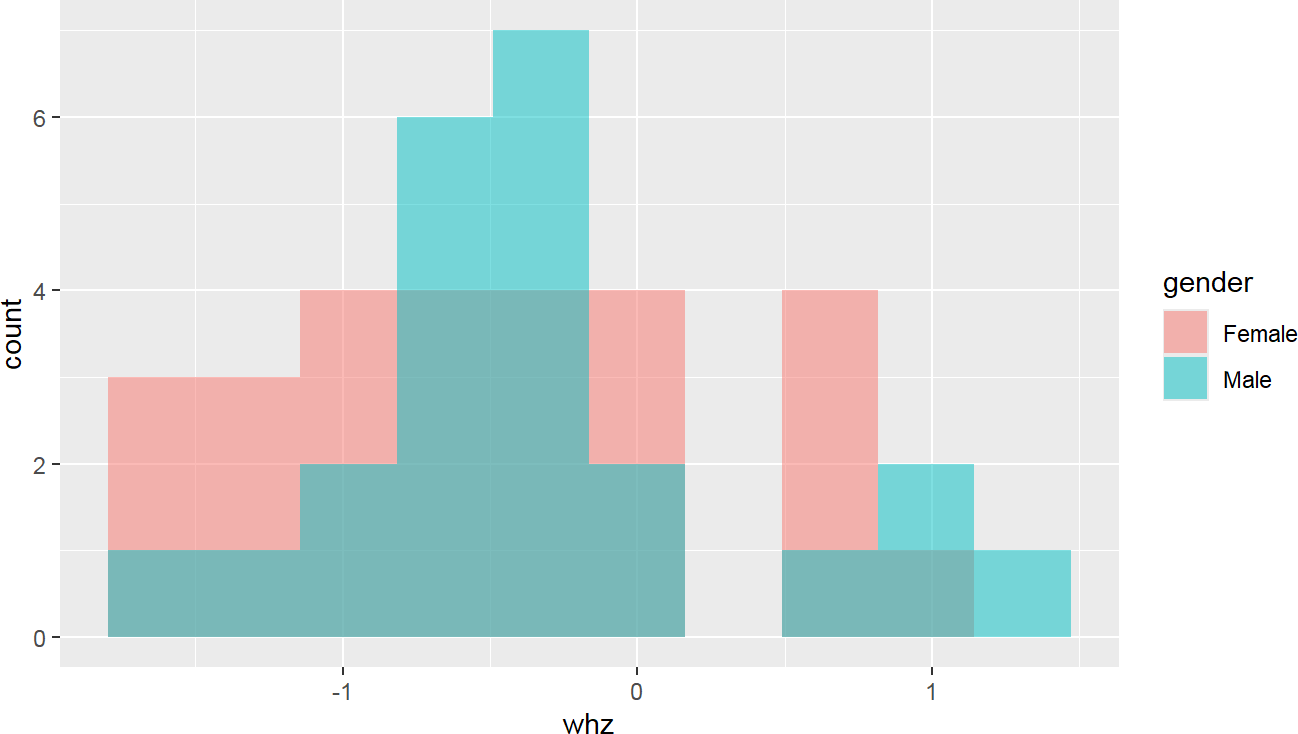


ggplot(dat2a,aes(whz))+ geom\_histogram(bins = 10)

Source: Figure by author(s).

**Figure 2**. Histogram example.

We can create a plot by group to compare the distribution across a categorical group (e.g. “gender”) by using the *fill* aesthetic.



ggplot(dat2a,aes(whz,fill = gender))+

geom\_histogram(bins = 10,alpha = 0.5, position = "identity")

Source: Figure by author(s).

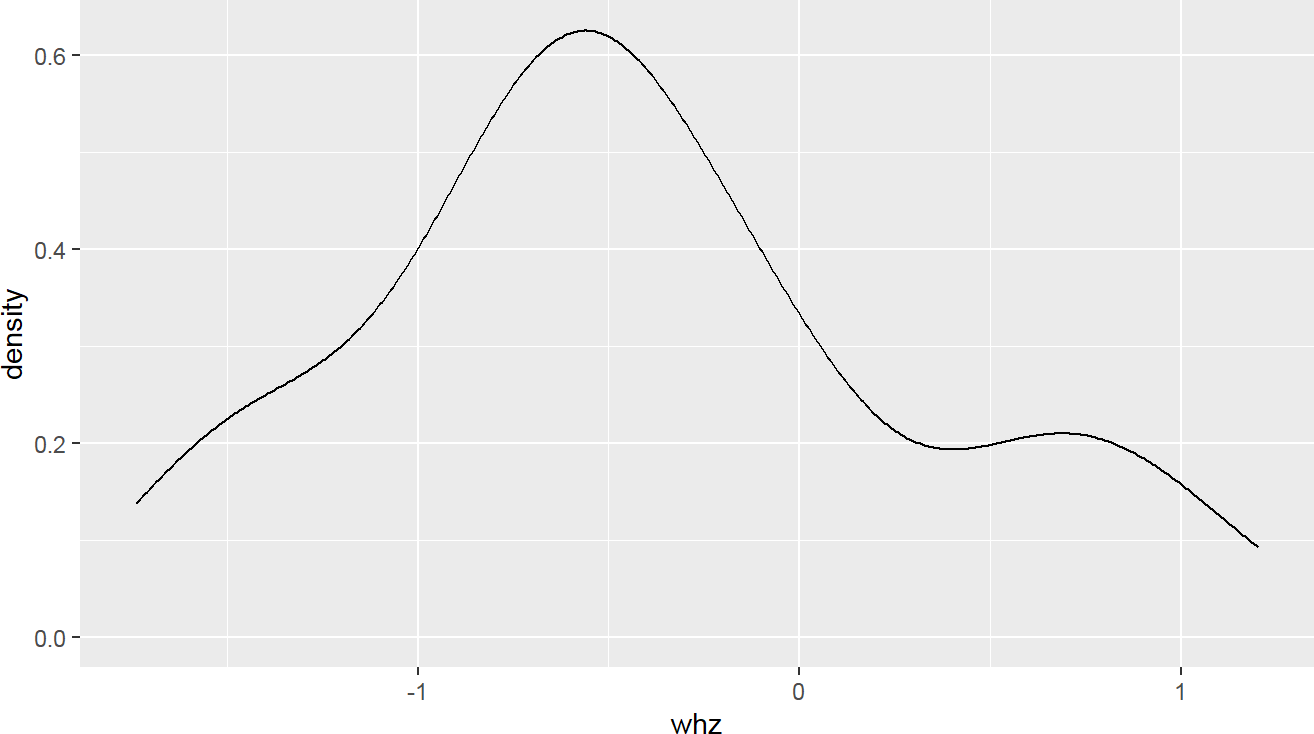
**Figure 3**. Plot by group example.

### Density

We can create a density plot with *geom\_density()*:

ggplot(dat2a, aes(whz)) +

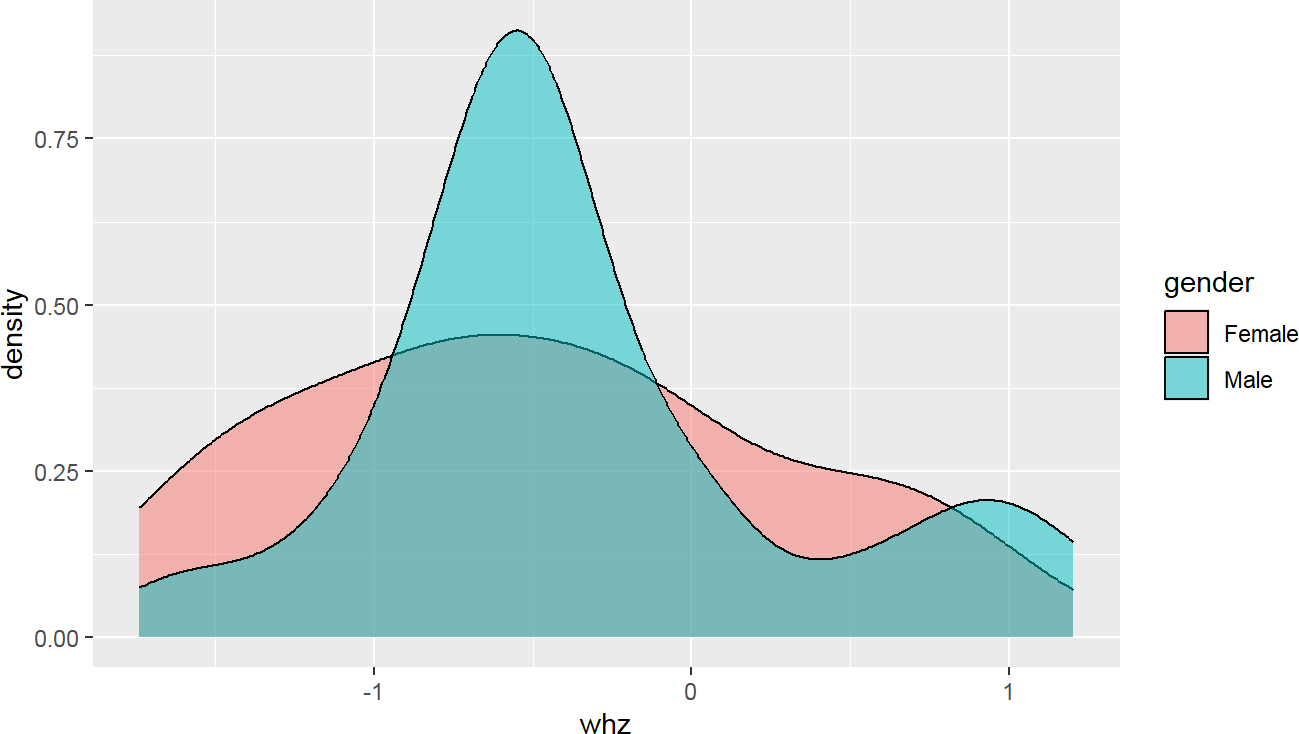
geom\_density()



Source: Figure by author(s).

**Figure 4**. Density example.

Similar to the above, we can create a plot by group using *fill*.



ggplot(dat2a, aes(whz, fill = gender)) +

geom\_density(alpha = 0.5)

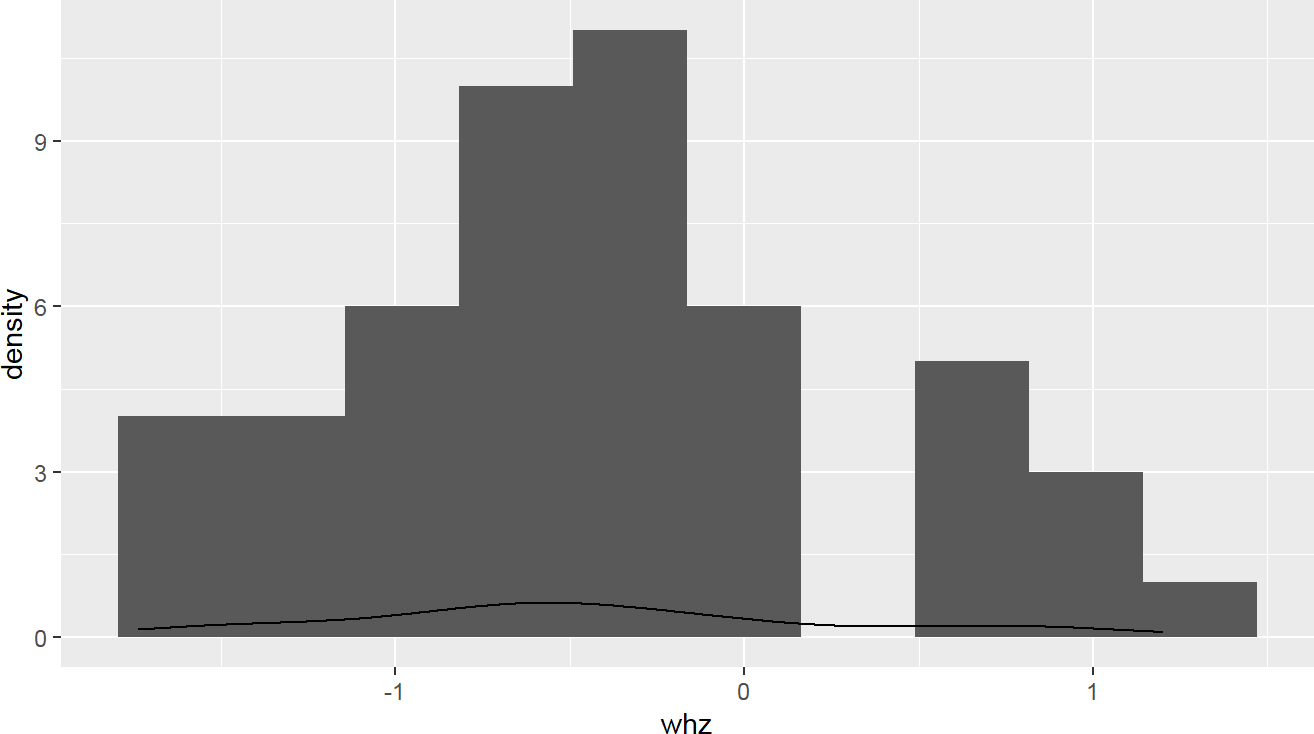
Source: Figure by author(s).

**Figure 5**. Density by group example.

### Histogram with Density

We can combine histogram and density into one plot:

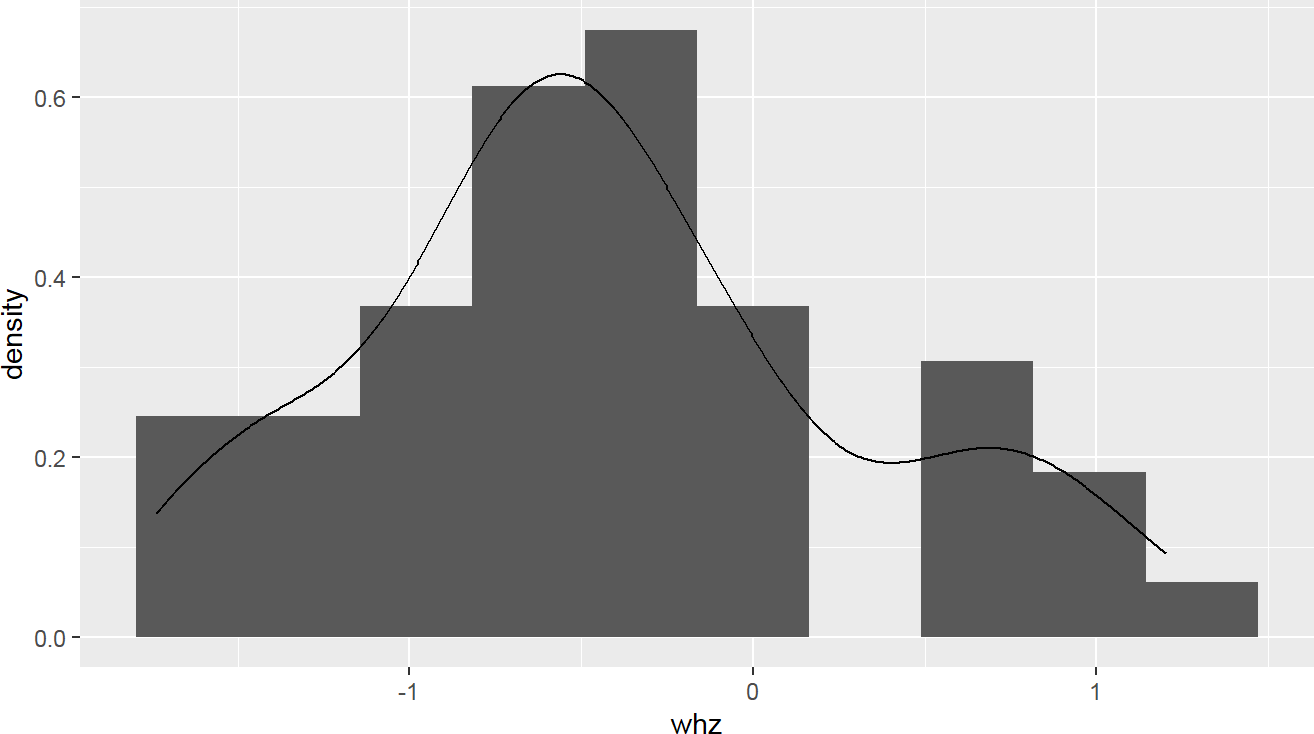
ggplot(dat2a,aes(whz))+ geom\_histogram(bins = 10)+ geom\_density()



Source: Figure by author(s).

**Figure 6**. Histogram with density example.

Something goes wrong! The density curve is on a different scale than the count-based histogram. To fix this, map the y-axis of the histogram to density using the computed variable *..density..*.:



ggplot(dat2a)+

geom\_histogram(aes(x = whz,y = ..density..),bins = 10)+

geom\_density(aes(whz))

Source: Figure by author(s).

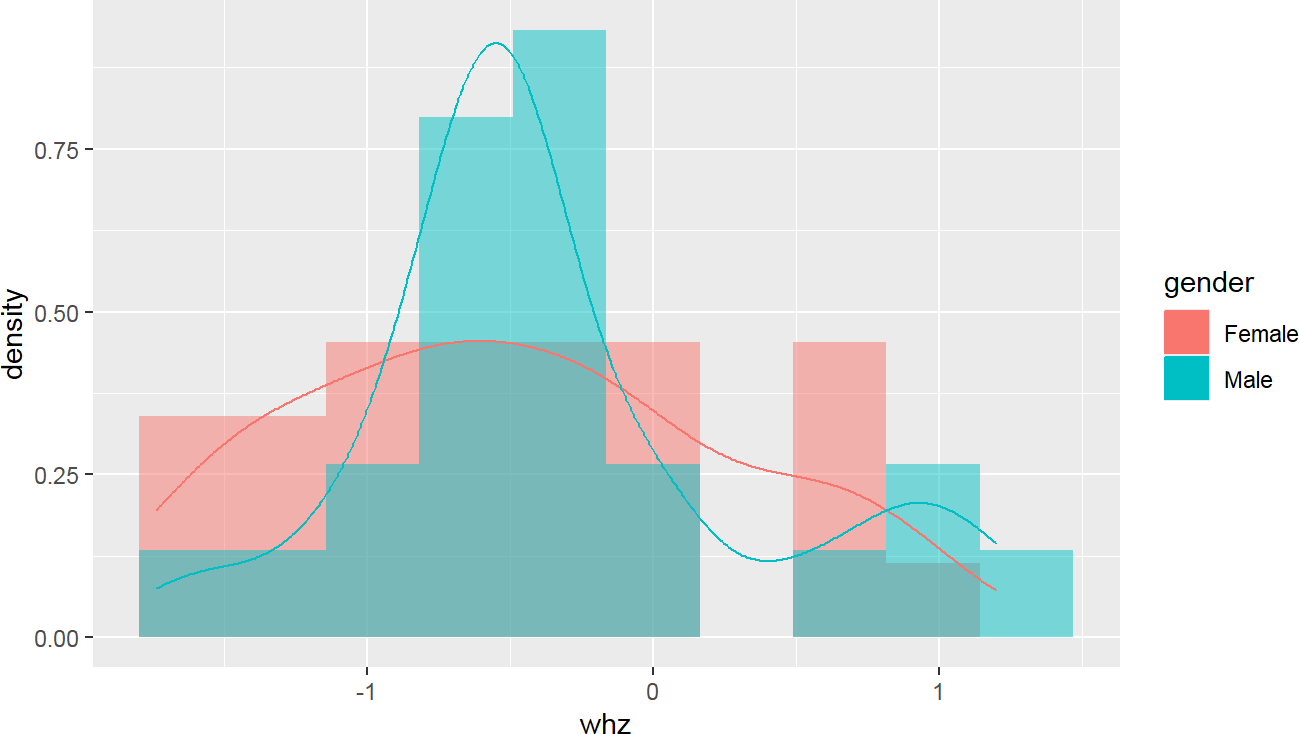
**Figure 7**. Histogram with density example fixed.

Again, we can create a plot by group using *fill*:

ggplot(dat2a)+

geom\_histogram(aes(x = whz,y = ..density..,fill = gender), bins = 10, position = "identity",alpha = 0.5)+

geom\_density(aes(whz,color = gender))

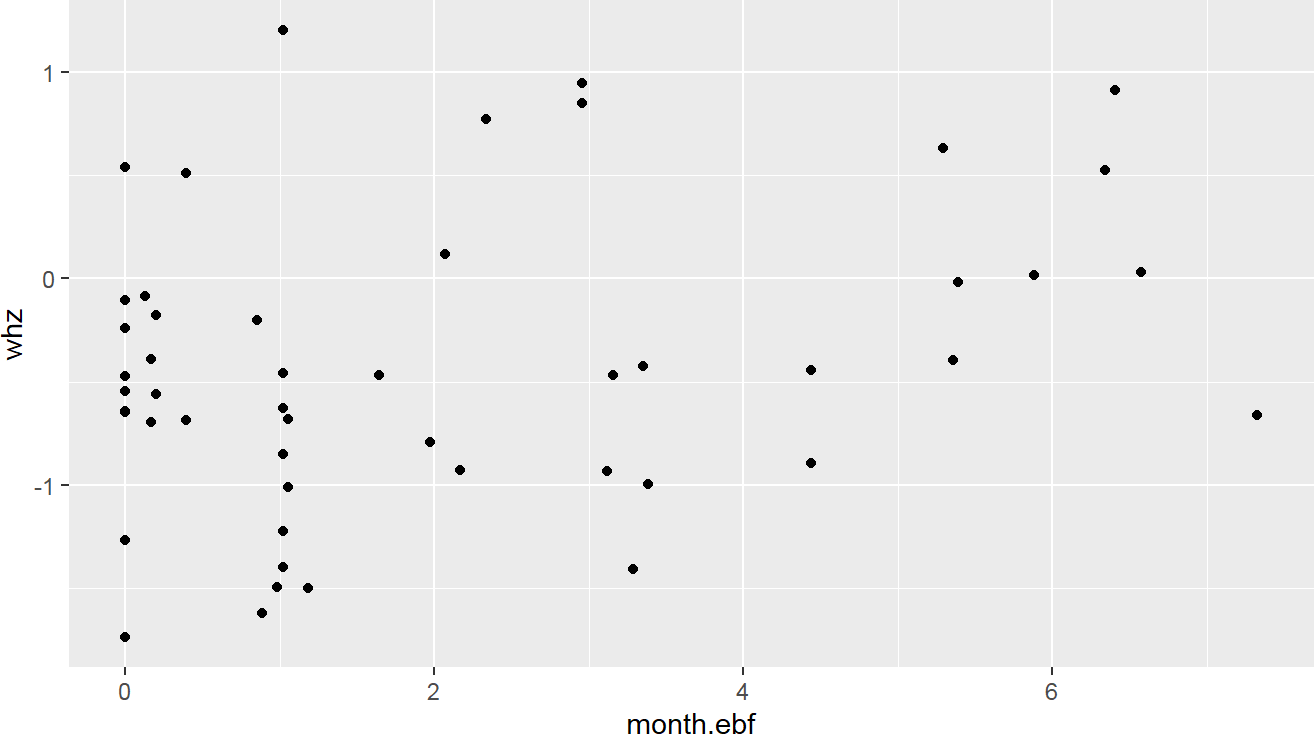


Source: Figure by author(s).

**Figure 8**. Histogram with density by group.

### Scatter Plot

We can create scatter plots with *geom\_point()*:



ggplot(dat2a,aes(x = month.ebf, y = whz)) +

geom\_point()

Source: Figure by author(s).

**Figure 9**. Scatter plot example.

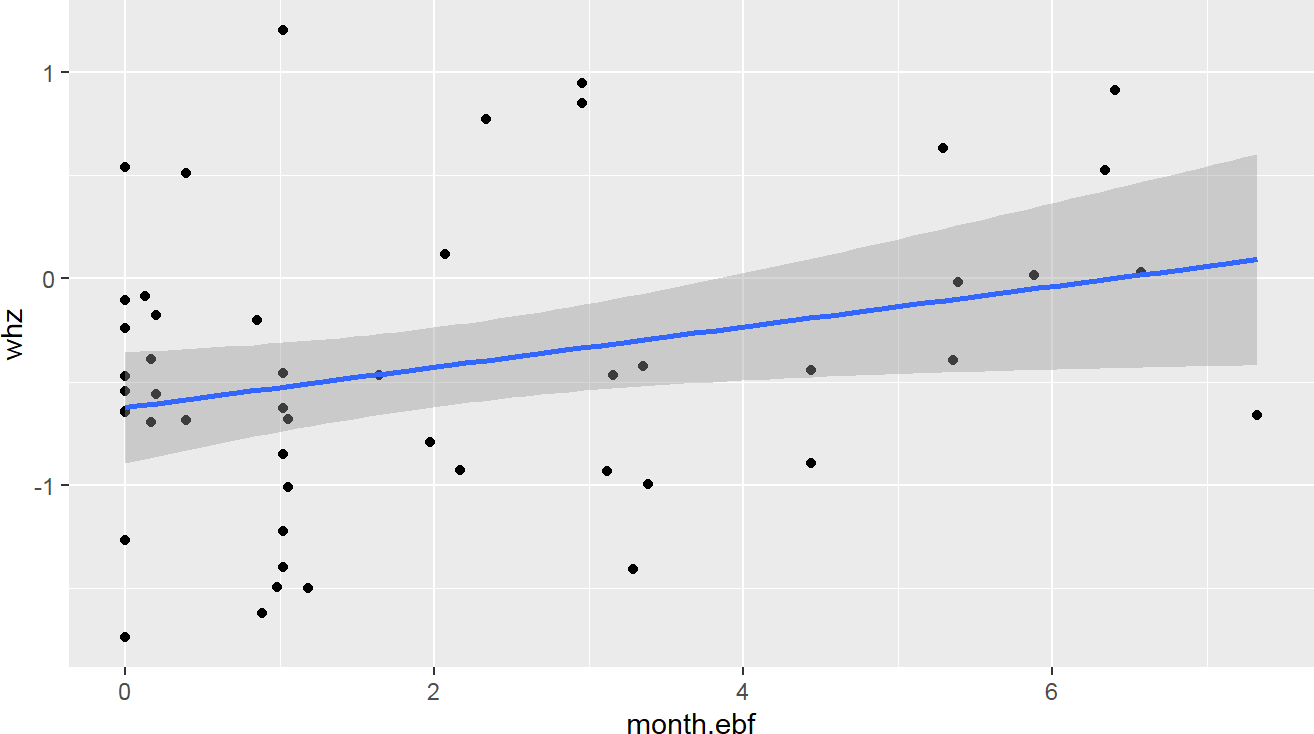
With Regression Line

A smoothed line (e.g., a regression line) can be added to summarize the trend. *stat\_smooth(method = 'glm')* adds a straight line (linear model, General Linear Model or *glm* is used here) representing the fitted relationship, along with its standard error (the shaded area).

ggplot(dat2a,aes(x = month.ebf, y = whz)) +

geom\_point()+

stat\_smooth(method = 'glm')



Source: Figure by author(s).

**Figure 10**. Scatter plot with regression line.

By group: in the example below, the color aesthetic is mapped to “gender”, creating separate colors and separate regression lines for each group.



ggplot(dat2a,aes(x = month.ebf, y = whz, color = gender)) +

geom\_point()+

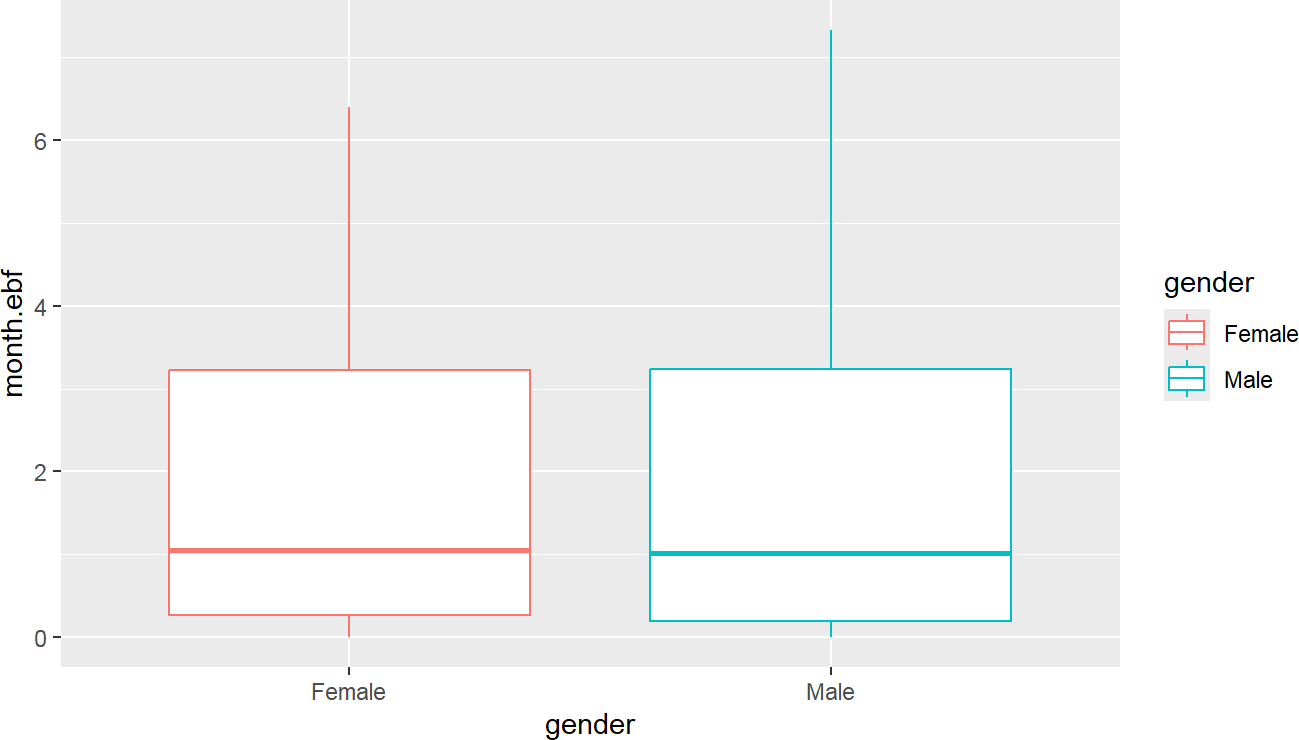
stat\_smooth(method = 'glm')

Source: Figure by author(s).

**Figure 11**. Scatter plot by group.

### Boxplot

We can create boxplots with *geom\_boxplot()*:



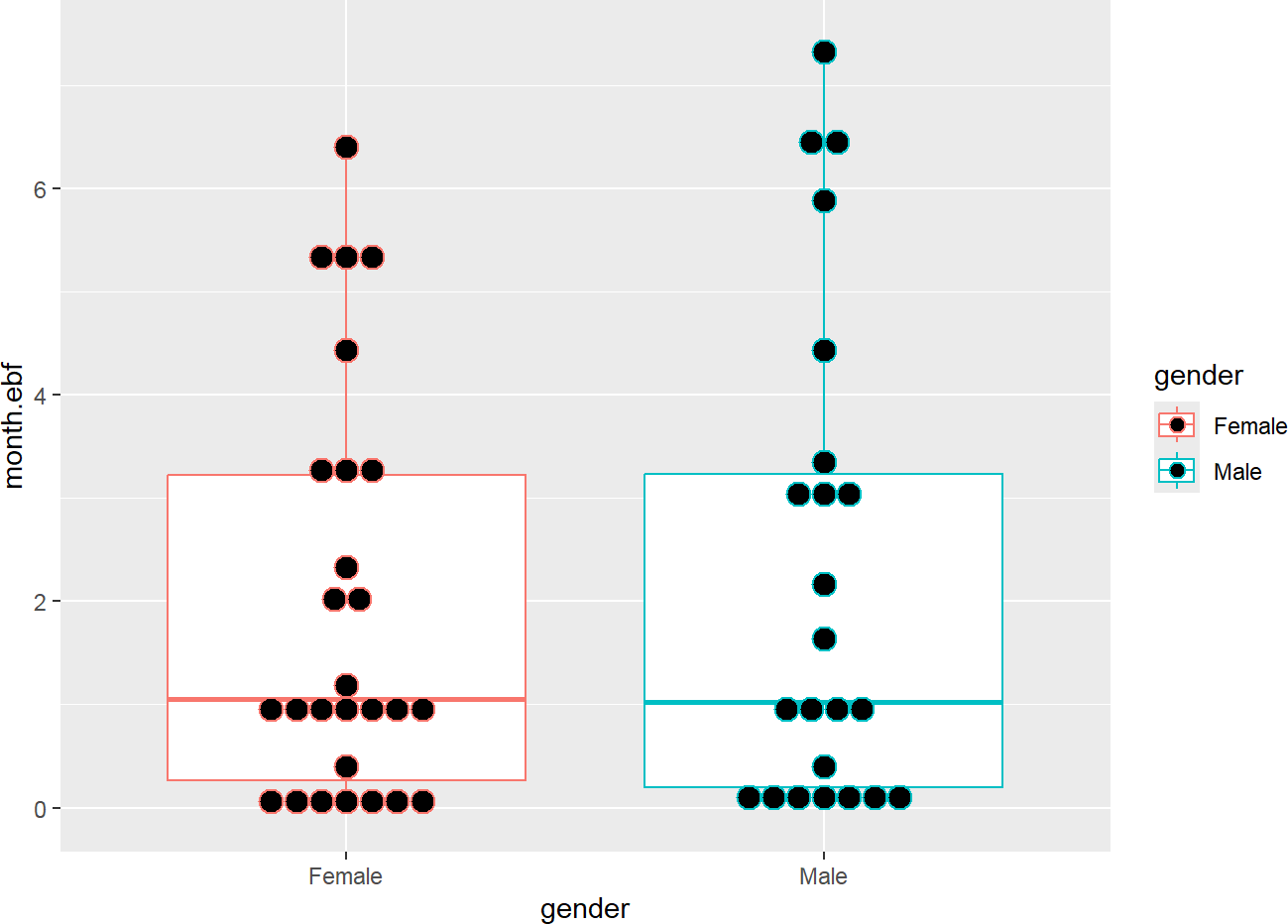
ggplot(dat2a,aes(x = gender, y = month.ebf, color = gender)) +

geom\_boxplot()

Source: Figure by author(s).

**Figure 12**. Boxplot example.

Box Plot with Dots: *geom\_dotplot()* plots the individual data points, stacking them (*stackdir = 'center'*) along the y-axis (*binaxis = 'y'*).



ggplot(dat2a,aes(x = gender, y = month.ebf, color = gender)) +

geom\_boxplot()+

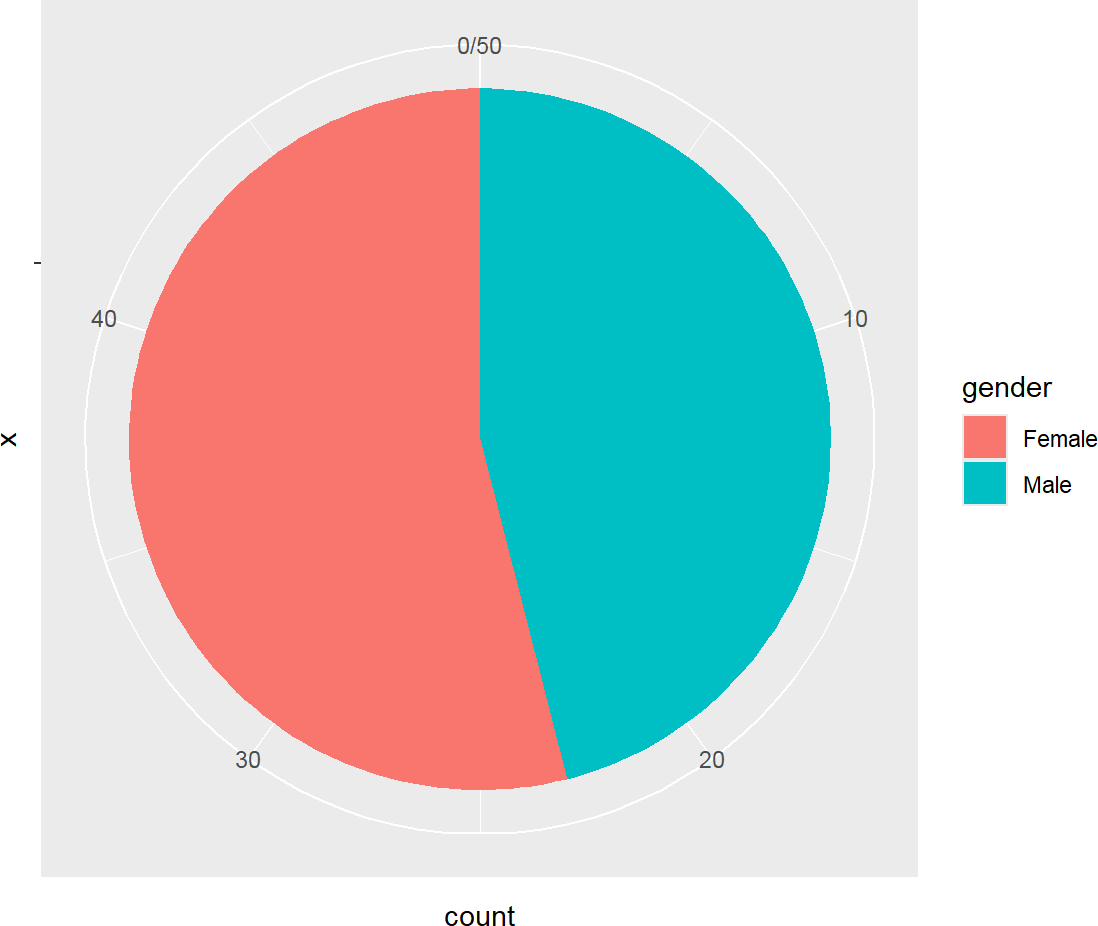
geom\_dotplot(binaxis = 'y', stackdir = 'center', dotsize = 1)

Source: Figure by author(s).

**Figure 13**. Box plot with dots.

### Pie Charts

In the example below, the data is grouped by gender and the count of each group is calculated (*summarise(count = n())*) then *ggplot()* creates the plot using the previously generated data from *summarise()* and *geom\_bar(stat = "identity") + coord\_polar("y", start = 0)* to generate pie plot. The *summarise()* function from *“dplyr”* is the primary tool for computing descriptive statistics, often used with *group\_by()* for group-level summaries.



dat2a %>%

group\_by(gender) %>%

summarise(count = n()) %>%

ggplot(aes(x = "",y = count,fill = gender)) +

geom\_bar(stat = "identity") +

coord\_polar("y", start = 0)

Source: Figure by author(s).

**Figure 14**. Pie Charts.

For more data visualization, below are some useful resources:

* + - * *“ggplot2”* tutorial ([http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-](http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html) [R-Code.html](http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html))
      * Statistical tools for high-throughput data analysis: *“ggplot2”* essentials ([http:](http://www.sthda.com/english/wiki/ggplot2-essentials)

[//www.sthda.com/english/wiki/ggplot2-essentials](http://www.sthda.com/english/wiki/ggplot2-essentials))

* + - * *“ggplot2”* cheat sheet ([https://rstudio.com/wp-content/uploads/2015/03/](https://rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf) [ggplot2-cheatsheet.pdf](https://rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf))

## Summary Statistics

### Basic Descriptive Summary Functions

The *summarise()* function from “*dplyr”* is the primary tool for computing descriptive statistics, often used with *group\_by()* for group-level summaries [17].

*# Summary of variable month.ebf*

dat2a %>% summarise(

count = n(),

mean = mean(month.ebf, na.rm = TRUE),

sd = sd(month.ebf, na.rm = TRUE),

median = median(month.ebf, na.rm = TRUE),

IQR = IQR(month.ebf, na.rm = TRUE))

## count mean sd median IQR ## 1 50 2.071644 2.154823 1.034836 3.055231

We can do summary by group with *group\_by()*:

*# Summary of month.ebf by cohort*

dat2a %>% group\_by(cohort) %>%

summarise(

count = n(),

mean = mean(month.ebf, na.rm = TRUE),

sd = sd(month.ebf, na.rm = TRUE),

median = median(month.ebf, na.rm = TRUE),

IQR = IQR(month.ebf, na.rm = TRUE))

## # A tibble: 2 x 6

## cohort count mean sd median IQR

## <chr> <int> <dbl> <dbl> <dbl> <dbl>

## 1 Healthy Singleton Birth Cohort 25 3.45 2.24 3.29 3.71

## 2 Healthy Twins & Triplets 25 0.694 0.706 0.854 1.02

We can also do summary by subgroup of another group (e.g. by “gender” within “Healthy Twins & Triplets”):

dat2a %>% filter(cohort == "Healthy Twins & Triplets") %>%

group\_by(gender) %>%

summarise(

count = n(),

mean = mean(month.ebf, na.rm = TRUE),

sd = sd(month.ebf, na.rm = TRUE),

median = median(month.ebf, na.rm = TRUE),

IQR = IQR(month.ebf, na.rm = TRUE))

## # A tibble: 2 x 6

## gender count mean sd median IQR

## <chr> <int> <dbl> <dbl> <dbl> <dbl>

## 1 Female 18 0.772 0.770 0.920 1.04

## 2 Male 7 0.493 0.498 0.197 0.920

We can add *filter()* to select specified rows:

dat2a %>% filter(cohort == "Healthy Twins & Triplets") %>%

filter(!is.na(zygosity)) %>%

group\_by(gender) %>%

summarise(

count = n(),

mean = mean(month.ebf, na.rm = TRUE),

sd = sd(month.ebf, na.rm = TRUE),

median = median(month.ebf, na.rm = TRUE),

IQR = IQR(month.ebf, na.rm = TRUE))

## # A tibble: 2 x 6

## gender count mean sd median IQR

## <chr> <int> <dbl> <dbl> <dbl> <dbl>

## 1 Female 16 0.749 0.816 0.690 1.03

## 2 Male 7 0.493 0.498 0.197 0.920

### Basic Statistical Test

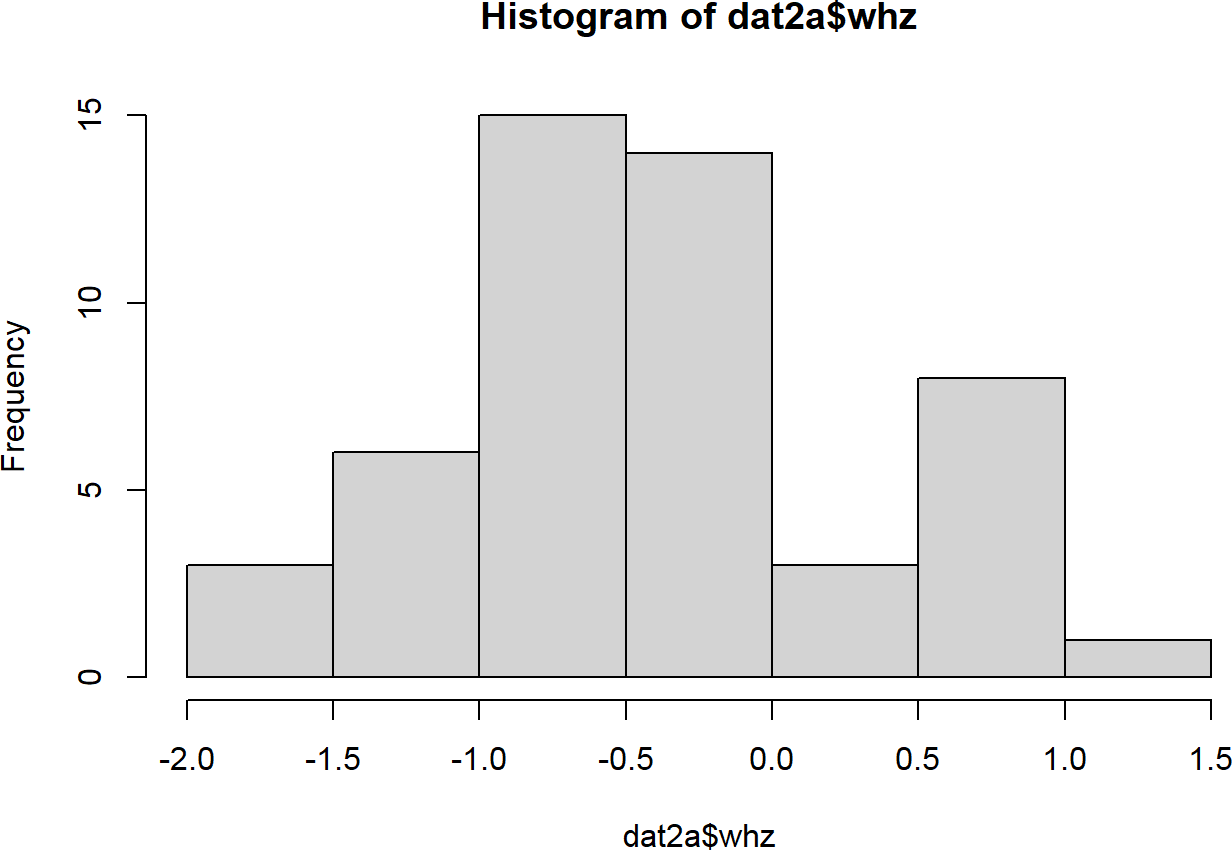
The example below uses parametric t-test to compare continuous variables between two groups:

*#t-test*

summary(dat2a$whz)

## Min. 1st Qu. Median Mean 3rd Qu. Max. ## -1.73842 -0.88598 -0.47063 -0.42223 -0.03458 1.20133

hist(dat2a$whz)



Source: Figure by author(s).

**Figure 15**. Histogram with base R plot.

t.test(whz~gender, data = dat2a)

##

## Welch Two Sample t-test

##

## data: whz by gender

## t = -0.7285, df = 47.802, p-value = 0.4699

## alternative hypothesis: true difference in means between group Female and group Male is not equal to 0

## 95 percent confidence interval:

## -0.5572199 0.2608470

## sample estimates:

## mean in group Female mean in group Male

## -0.4903957 -0.3422092

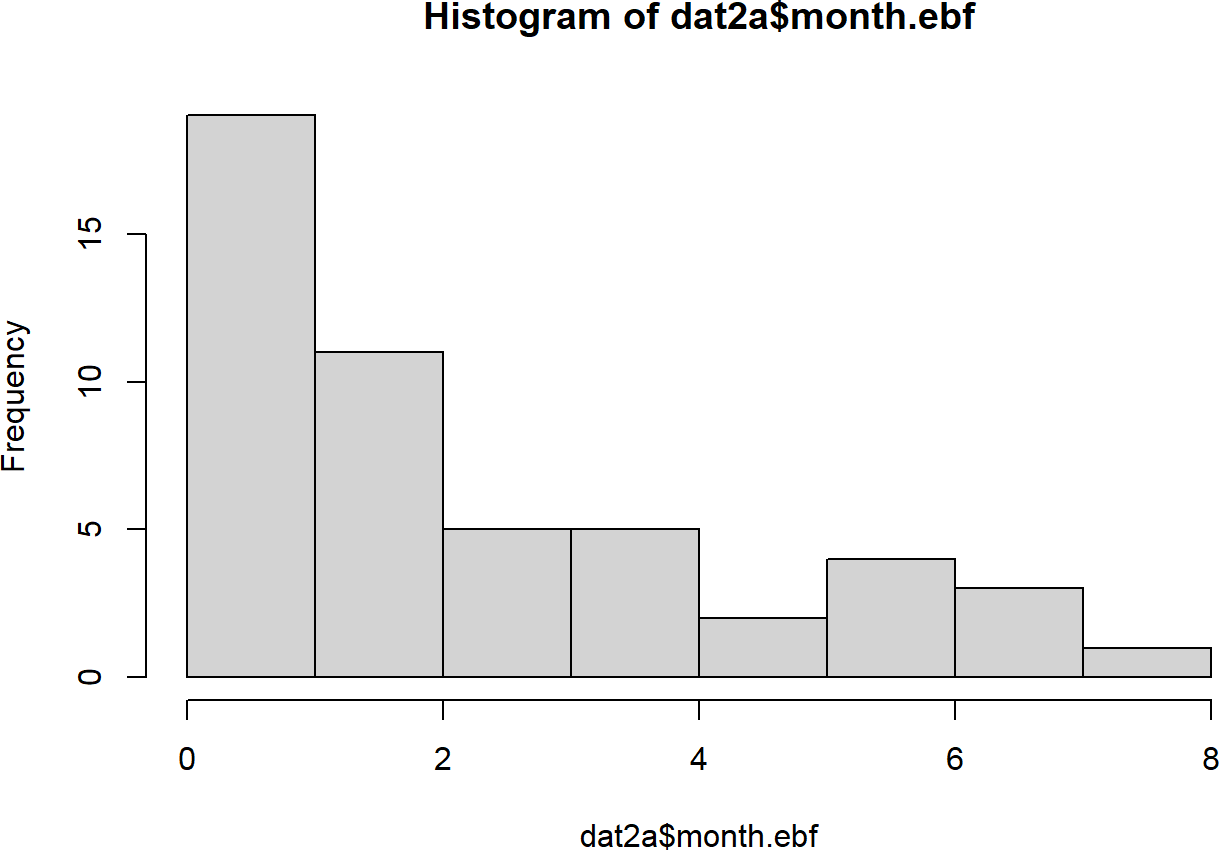
The example below uses Non-Parametric (Wilcoxon or Kruskal-Wallis) test to compare continuous variable:

*#Kruskal-Wallis*

summary(dat2a$month.ebf)

## Min. 1st Qu. Median Mean 3rd Qu. Max. ## 0.0000 0.1971 1.0348 2.0716 3.2523 7.3260

hist(dat2a$month.ebf)



Source: Figure by author(s).

**Figure 16**. Histogram with base R plot 2.

kruskal.test(month.ebf ~ gender, data = dat2a)

##

## Kruskal-Wallis rank sum test

##

## data: month.ebf by gender

## Kruskal-Wallis chi-squared = 0.12372, df = 1, p-value = ## 0.725

*#wilcox.test*

wilcox.test(month.ebf ~ gender, data = dat2a)

##

## Wilcoxon rank sum test with continuity correction

##

## data: month.ebf by gender

## W = 292.5, p-value = 0.7324

## alternative hypothesis: true location shift is not equal to 0

The examples below use Chi-square and Fisher’s exact tests to compare categorical variables:

|  |  |  |
| --- | --- | --- |
| *#Chi-square*  table(dat2a$cohort,dat2a$gender) |  |  |
| ## ## | Female | Male |
| ## Healthy Singleton Birth Cohort | 9 | 16 |
| ## Healthy Twins & Triplets | 18 | 7 |

chisq.test(dat2a$cohort,dat2a$gender)

##

## Pearson's Chi-squared test with Yates' continuity

## correction

##

## data: dat2a$cohort and dat2a$gender

## X-squared = 5.153, df = 1, p-value = 0.02321

*#Fisher*

fisher.test(dat2a$cohort,dat2a$gender)

##

## Fisher's Exact Test for Count Data

##

## data: dat2a$cohort and dat2a$gender

## p-value = 0.02224

## alternative hypothesis: true odds ratio is not equal to 1

## 95 percent confidence interval:

## 0.05544835 0.83324786

## sample estimates:

## odds ratio

## 0.2261644

### Summary Statistic Table

#### Packages for Quick, Nice Summary Table

The “*arsenal*” package provides handy R functions for large-scale statistical summaries. The *tableby()* function of the arsenal package is used to create a summary table that includes basic statistical tests for comparing many variables between groups [19].

In the example below, the *tableby()* function is applied to the Bangladesh children dataset above using a formula like “*group variable ~ variables to be summarized*” to summarize those variables stratified by the cohort group and perform statistical tests for difference. The packages "*tidyverse*" [11] for data handling and visualization, "*knitr*" [5], and "*kableExtra*" [12] for making tables will use throughout the chapters of this book.

library(arsenal)

ca.vars<-c("cohort","gender","zygosity")

co.vars<-c("whz", "waz", "haz","age.first.fe","age.last.fe", "no.fe.sam","sam.int.mean",

"month.ebf","age.first.solid","no.diarhea.yr", "pc.days.diarrhea","frac.sam.atb.7day")

varuse<-c(co.vars,ca.vars[!ca.vars %in% "cohort"]) mylabels <-as.list(varuse)

names(mylabels)<-varuse

tab1 <- tableby(as.formula(paste("cohort",paste(varuse,collapse = "+"),sep = "~")), data = dat2a)

kable(summary(tab1,

labelTranslations = mylabels, text = TRUE),

caption = "Summary table")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 3**. Summary table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Healthy Singleton Birth Cohort (N = 25)** | **Healthy Twins & Triplets (N = 25)** | **Total (N = 50)** | ***p* Value** |
| whz |  |  |  | 0.683 |
| - Mean (SD) | −0.380 (0.727) | −0.464 (0.724) | −0.422 (0.719) |  |
| - Range | −1.623–0.949 | −1.738–1.201 | −1.738–1.201 |  |
| waz |  |  |  | <0.001 |
| - Mean (SD) | −0.483 (0.806) | −2.546 (0.561) | −1.515 (1.248) |  |
| - Range | −1.741–1.670 | −3.755–1.392 | −3.755–1.670 |  |
| haz |  |  |  | <0.001 |
| - Mean (SD) | −0.483 (0.806) | −3.231 (0.956) | −1.857 (1.641) |  |
| - Range | −1.741–1.670 | −4.659–1.412 | −4.659–1.670 |  |
| age.first.fe |  |  |  | 0.008 |
| - Mean (SD) | 6.360 (2.782) | 13.200 (12.124) | 9.780 (9.366) |  |
| - Range | 3.000–15.000 | 1.000–37.000 | 1.000–37.000 |  |
| age.last.fe |  |  |  | <0.001 |
| - Mean (SD) | 710.400 (12.430) | 490.880 (152.553) | 600.640 (154.167) |  |
| - Range | 700.000–738.000 | 286.000–730.000 | 286.000–738.000 |  |
| no.fe.sam |  |  |  | 0.001 |
| - Mean (SD) | 21.920 (2.019) | 17.920 (5.338) | 19.920 (4.476) |  |
| - Range | 18.000–24.000 | 9.000–26.000 | 9.000–26.000 |  |
| sam.int.mean |  |  |  | <0.001 |
| - Mean (SD) | 34.003 (3.813) | 28.677 (3.090) | 31.340 (4.363) |  |
| - Range | 30.087–42.294 | 22.562–35.000 | 22.562–42.294 |  |
| month.ebf |  |  |  | < 0.001 |
| - Mean (SD) | 3.449 (2.242) | 0.694 (0.706) | 2.072 (2.155) |  |
| - Range | 0.000–7.326 | 0.000–2.332 | 0.000–7.326 |  |
| age.first.solid |  |  |  | 0.254 |
| - Mean (SD) | 6.352 (2.236) | 7.028 (1.884) | 6.690 (2.074) |  |
| - Range | 1.216–9.790 | 3.088–9.396 | 1.216–9.790 |  |
| no.diarhea.yr |  |  |  | <0.001 |
| - Mean (SD) | 4.278 (2.135) | 1.664 (1.345) | 2.971 (2.205) |  |
| - Range | 0.520–8.814 | 0.000–4.986 | 0.000–8.814 |  |
| pc.days.diarrhea |  |  |  | 0.007 |
| - Mean (SD) | 4.090 (2.223) | 2.159 (2.601) | 3.125 (2.585) |  |
| - Range | 0.142–10.826 | 0.000–9.563 | 0.000–10.826 |  |
| frac.sam.atb.7day |  |  |  | 0.070 |
| - Mean (SD) | 0.219 (0.145) | 0.148 (0.125) | 0.183 (0.139) |  |
| - Range | 0.000–0.500 | 0.000–0.462 | 0.000–0.500 |  |

**Table 3.** *Cont.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Healthy Singleton Birth Cohort (N = 25)** | **Healthy Twins &**  **Triplets (N = 25) Total (N = 50) *p* Value** | | |
| gender |  |  |  | 0.011 |
| - Female | 9 (36.0%) | 18 (72.0%) | 27 (54.0%) |  |
| - Male | 16 (64.0%) | 7 (28.0%) | 23 (46.0%) |  |
| zygosity |  |  |  | <0.001 |
| - N-Miss | 0 | 2 | 2 |  |
| - DZ | 0 (0.0%) | 12 (52.2%) | 12 (25.0%) |  |
| - Fraternal co-twin in set of triplets | 0 (0.0%) | 1 (4.3%) | 1 (2.1%) |  |
| - MZ | 0 (0.0%) | 8 (34.8%) | 8 (16.7%) | |
| - MZ co-twin in set of 0 (0.0%) 2 (8.7%) 2 (4.2%) | | | | |
| triplets |  |  |  | |
| - singleton | 25 (100.0%) | 0 (0.0%) | 25 (52.1%) | |

Source: Table by author(s).

Readers feel free to try other R packages for nice data summary tables.

#### More Comprehensive Table (Control)

The *tableby.control()* function allows fine-tuning of the output, such as changing the statistical tests used (e.g., using a Kruskal Wallis test (kwt) instead of t-test) and adjusting the summary statistics displayed (e.g., adding Mean (95%CI)).

my\_controls <- tableby.control( test = T,

total = T,

numeric.test = "kwt", cat.test = "fe",

numeric.stats = c("meansd", "medianq1q3", "range", "Nmiss2"), cat.stats = c("countpct", "Nmiss2"),

stats.labels = list( meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)", range = "Min - Max",

Nmiss2 = "Missing"

)

)

tab2 <- tableby(as.formula(paste("cohort",paste(varuse,collapse = "+"),sep = "~")), data = dat2a,

control = my\_controls) kable(summary(tab2,labelTranslations = mylabels, text = TRUE),

caption = "More comprehensive summary table")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 4**. More comprehensive summary table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Healthy Singleton Birth Cohort (N = 25)** | **Healthy Twins & Triplets (N = 25)** | **Total (N = 50)** | ***p* Value** |
| whz |  |  |  | 0.705 |
| - Mean (SD) | −0.380 (0.727) | −0.464 (0.724) | −0.422 (0.719) |  |
| - Median (Q1, Q3) | −0.468 (−0.897, 0.015) | −0.560 (−0.853, −0.104) | −0.471 (−0.886, −0.035) |  |
| - Min−Max | −1.623−0.949 | −1.738−1.201 | −1.738−1.201 |  |
| - Missing | 0 | 0 | 0 |  |
| waz |  |  |  | <0.001 |
| - Mean (SD) | −0.483 (0.806) | −2.546 (0.561) | −1.515 (1.248) |  |
| - Median (Q1, Q3) | −0.706 (−1.031, −0.030) | −2.606 (−2.879, −2.295) | −1.512 (−2.604, −0.719) |  |
| - Min−Max | −1.741−1.670 | −3.755−−1.392 | −3.755−1.670 |  |
| - Missing | 0 | 0 | 0 |  |
| haz |  |  |  | <0.001 |
| - Mean (SD) | −0.483 (0.806) | −3.231 (0.956) | −1.857 (1.641) |  |
| - Median (Q1, Q3) | −0.706 (−1.031, −0.030) | −3.437 (−4.032, −2.551) | −1.575 (−3.431, −0.719) |  |
| - Min−Max | −1.741−1.670 | −4.659−−1.412 | −4.659−1.670 |  |
| - Missing | 0 | 0 | 0 |  |
| age.first.fe |  |  |  | 0.077 |
| - Mean (SD) | 6.360 (2.782) | 13.200 (12.124) | 9.780 (9.366) |  |
| - Median (Q1, Q3) | 5.000 (5.000, 8.000) | 8.000 (5.000, 15.000) | 6.000 (5.000, 9.750) |  |
| - Min−Max | 3.000−15.000 | 1.000−37.000 | 1.000−37.000 |  |
| - Missing | 0 | 0 | 0 |  |
| age.last.fe |  |  |  | < 0.001 |
| - Mean (SD) | 710.400 (12.430) | 490.880 (152.553) | 600.640 (154.167) |  |
| - Median (Q1, | 704.000 (701.000, | 455.000 (366.000, | 700.500 (455.000, |  |
| Q3) | 715.000) | 638.000) | 706.000) |  |
| - Min−Max | 700.000−738.000 | 286.000−730.000 | 286.000−738.000 |  |
| - Missing | 0 | 0 | 0 |  |
| no.fe.sam |  |  |  | 0.011 |
| - Mean (SD) | 21.920 (2.019) | 17.920 (5.338) | 19.920 (4.476) |  |
| - Median (Q1, Q3) | 23.000 (21.000, 23.000) | 18.000 (13.000, 22.000) | 21.000 (18.000, 23.000) |  |
| - Min−Max | 18.000−24.000 | 9.000−26.000 | 9.000−26.000 |  |
| - Missing | 0 | 0 | 0 |  |
| sam.int.mean |  |  |  | < 0.001 |
| - Mean (SD) | 34.003 (3.813) | 28.677 (3.090) | 31.340 (4.363) |  |
| - Median (Q1, Q3) | 33.143 (31.409, 35.350) | 28.154 (27.048, 30.350) | 30.477 (28.290, 33.271) |  |
| - Min−Max | 30.087−42.294 | 22.562−35.000 | 22.562−42.294 |  |
| - Missing | 0 | 0 | 0 |  |

**Table 4.** *Cont.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Healthy Singleton Birth Cohort (N = 25)** | **Healthy Twins & Triplets (N = 25)** | **Total (N = 50)** | ***p* Value** |
| month.ebf |  |  |  | < 0.001 |
| - Mean (SD) | 3.449 (2.242) | 0.694 (0.706) | 2.072 (2.155) |  |
| - Median (Q1, Q3) | 3.285 (1.643, 5.355) | 0.854 (0.000, 1.018) | 1.035 (0.197, 3.252) |  |
| - Min−Max | 0.000−7.326 | 0.000−2.332 | 0.000−7.326 |  |
| - Missing | 0 | 0 | 0 |  |
| age.first.solid |  |  |  | 0.388 |
| - Mean (SD) | 6.352 (2.236) | 7.028 (1.884) | 6.690 (2.074) |  |
| - Median (Q1, Q3) | 6.209 (4.599, 8.213) | 7.063 (6.078, 8.476) | 6.932 (4.920, 8.361) |  |
| - Min−Max | 1.216−9.790 | 3.088−9.396 | 1.216−9.790 |  |
| - Missing | 0 | 0 | 0 |  |
| no.diarhea.yr |  |  |  | < 0.001 |
| - Mean (SD) | 4.278 (2.135) | 1.664 (1.345) | 2.971 (2.205) |  |
| - Median (Q1, Q3) | 3.957 (3.008, 5.170) | 1.276 (0.802, 2.003) | 2.916 (1.271, 4.054) |  |
| - Min−Max | 0.520−8.814 | 0.000−4.986 | 0.000−8.814 |  |
| - Missing | 0 | 0 | 0 |  |
| pc.days.diarrhea |  |  |  | < 0.001 |
| - Mean (SD) | 4.090 (2.223) | 2.159 (2.601) | 3.125 (2.585) |  |
| - Median (Q1, Q3) | 3.930 (2.996, 4.755) | 1.359 (0.440, 2.740) | 2.742 (1.163, 4.314) |  |
| - Min−Max | 0.142−10.826 | 0.000− 9.563 | 0.000−10.826 |  |
| - Missing | 0 | 0 | 0 |  |
| frac.sam.atb.7day |  |  |  | 0.109 |
| - Mean (SD) | 0.219 (0.145) | 0.148 (0.125) | 0.183 (0.139) |  |
| - Median (Q1, Q3) | 0.217 (0.087, 0.318) | 0.111 (0.077, 0.167) | 0.119 (0.081, 0.284) |  |
| - Min−Max | 0.000−0.500 | 0.000−0.462 | 0.000−0.500 |  |
| - Missing | 0 | 0 | 0 |  |
| gender |  |  |  | 0.022 |
| - Female | 9 (36.0%) | 18 (72.0%) | 27 (54.0%) |  |
| - Male | 16 (64.0%) | 7 (28.0%) | 23 (46.0%) |  |
| - Missing | 0 | 0 | 0 |  |
| zygosity |  |  |  | < 0.001 |
| - DZ | 0 (0.0%) | 12 (52.2%) | 12 (25.0%) |  |
| - Fraternal  co-twin in set | 0 (0.0%) | 1 (4.3%) | 1 (2.1%) |  |
| of triplets |  |  |  |  |
| - MZ | 0 (0.0%) | 8 (34.8%) | 8 (16.7%) |  |
| - MZ co-twin in set of triplets | 0 (0.0%) | 2 (8.7%) | 2 (4.2%) |  |
| - singleton | 25 (100.0%) | 0 (0.0%) | 25 (52.1%) |  |
| - Missing | 0 | 2 | 2 |  |

Source: Table by author(s).

## Practice

Write R codes to:

* Import sheet 2 of excel data file “nature13421-s2.xlsx” introduced above
* Rename column names to sensible names, remove redundant rows
* Save your cleaned data to local disc and load saved data for use
* Make a table showing number of fecal sample per each child ID
* Summarize number of fecal samples in all child ID and by cohort of singletons

vs twins/triplets

* Plot number of fecal samples by cohort of singletons vs twins/triplets

## Practice Review

### The Requirements

We now practice writing R codes for the requirements above. First, we import sheet 2 of excel data file using the “*rio*” package [14]:

library(rio)

*#import all sheets*

list1<-import\_list(paste(fdir1,"nature13421-s2.xlsx",sep = ""))

*#get sheet 2*

sh2<-list1[[2]]

*#import sheet 2 only*

sh2<-import(paste(fdir1,"nature13421-s2.xlsx",sep = ""), sheet = 2,skip = 1)

We rename column names to sensible names, and remove redundant rows:

library(tidyverse)

*#colnames(sh2)*

colnames(sh2)[!is.na(sh2[1,])]<-sh2[1,][!is.na(sh2[1,])]

*#or just do all at a time* oldname<-colnames(sh2) newname<-c("cohort",

"family.id",

"child.id",

"sample.id",

"age.d",

"age.m", "whz","waz","haz", "breast.milk", "formula", "solid", "diarrhea", "antibiotic.7d", "medication", "no.sequence",

"serun.id", "barcode")

sh2<-sh2 %>% rename\_at(vars(oldname), ~newname)

*#colnames(sh2) #remove unused rows* sh2<-sh2 %>%

filter(!is.na(family.id))

You may want to save your cleaned data to local disc and load saved data for

use:

*#save data*

*#write.csv(sh2,file = paste(fdir1,"sh2.csv",sep = ""),row.names = F)*

*#sh2<-read.csv(paste(fdir1,"sh2.csv",sep = ""))*

Now, we make a table showing number of fecal sample per each child ID and summarize number of fecal samples in all child ID:

nsam<-sh2 %>% group\_by(child.id) %>% summarise(no.sam = n())

*#View(nsam)*

nsam %>%

summarise(

mean = mean(no.sam, na.rm = TRUE),

sd = sd(no.sam, na.rm = TRUE),

median = median(no.sam, na.rm = TRUE),

IQR = IQR(no.sam, na.rm = TRUE))

## # A tibble: 1 x 4

## mean sd median IQR

## <dbl> <dbl> <dbl> <dbl>

## 1 19.9 4.48 21 5

We summarize number of fecal samples per “child.id” by cohort of singletons vs twins/triplets:

nsam<-sh2 %>%

group\_by(cohort,child.id) %>%

summarise(no.sam = n()) %>%

summarise(

mean = mean(no.sam, na.rm = TRUE),

sd = sd(no.sam, na.rm = TRUE),

median = median(no.sam, na.rm = TRUE),

IQR = IQR(no.sam, na.rm = TRUE))

nsam

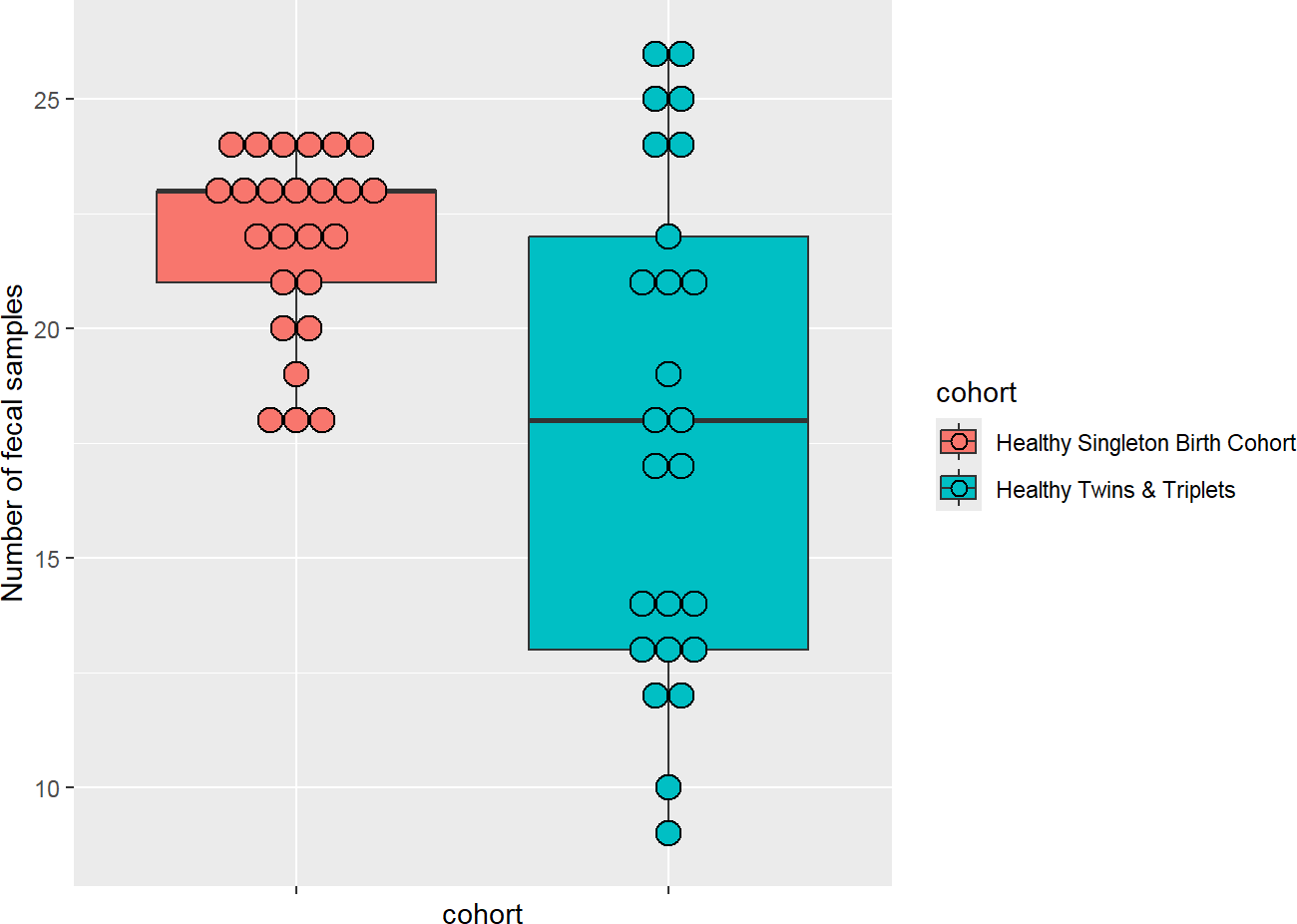
## # A tibble: 2 x 5

## cohort mean sd median IQR

## <chr> <dbl> <dbl> <int> <dbl> ## 1 Healthy Singleton Birth Cohort 21.9 2.02 23 2

## 2 Healthy Twins & Triplets 17.9 5.34 18 9

We create a boxplot for number of fecal samples by cohort of singletons vs twins/triplets:



sh2 %>%

group\_by(cohort,child.id) %>% summarise(no.sam = n()) %>%

ggplot(aes(x = cohort,y = no.sam,fill = cohort)) +

geom\_boxplot()+

ylab("Number of fecal samples")+

geom\_dotplot(binaxis = 'y', stackdir = 'center', dotsize = 1)+

theme(axis.text.x = element\_blank())

Source: Figure by author(s).

**Figure 17**. Boxplot number of fecal samples.

We can create a violin plot which is similar to box plot but shows the density within groups (not much information provided as in box plot):

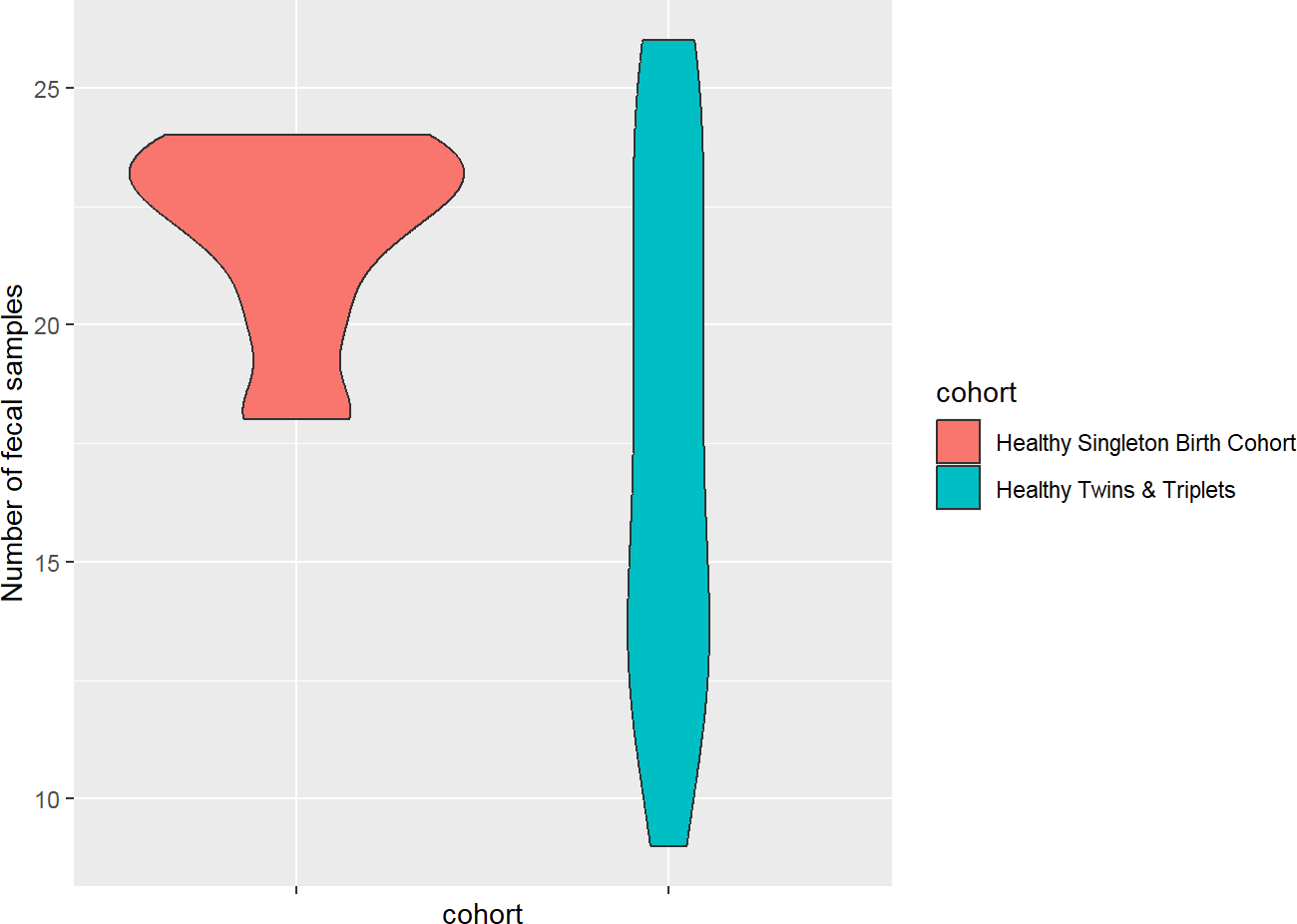
sh2 %>% group\_by(cohort,child.id) %>% summarise(no.sam = n()) %>%

ggplot(aes(x = cohort,y = no.sam,fill = cohort)) +

geom\_violin()+

ylab("Number of fecal samples")+

theme(axis.text.x = element\_blank())



Source: Figure by author(s).

**Figure 18**. Violin plot number of fecal samples.

We can create a bar plot for number of fecal samples by cohort of singletons vs. twins/triplets and a summary of number of fecal samples:

nsam<-sh2 %>%

group\_by(cohort,child.id) %>% summarise(no.sam = n()) %>%

summarise(

no.child = n(),

mean = mean(no.sam, na.rm = TRUE),

sd = sd(no.sam, na.rm = TRUE),

se = sd/sqrt(n()),

ci = 1.96\*se,

median = median(no.sam, na.rm = TRUE),

IQR = IQR(no.sam, na.rm = TRUE))

nsam

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ##  ## | # | A tibble: 2 x 8  cohort | no.child | mean | sd | se | ci | median | IQR |
| ## |  | <chr> | <int> | <dbl> | <dbl> | <dbl> | <dbl> | <int> | <dbl> |
| ## | 1 | Healthy Singleto~ | 25 | 21.9 | 2.02 | 0.404 | 0.791 | 23 | 2 |
| ## | 2 | Healthy Twins & ~ | 25 | 17.9 | 5.34 | 1.07 | 2.09 | 18 | 9 |

We can create a bar plot with error bar:

ggplot(nsam, aes(x = cohort, y = mean, fill = cohort)) +

geom\_bar(position = position\_dodge(), stat = "identity") +

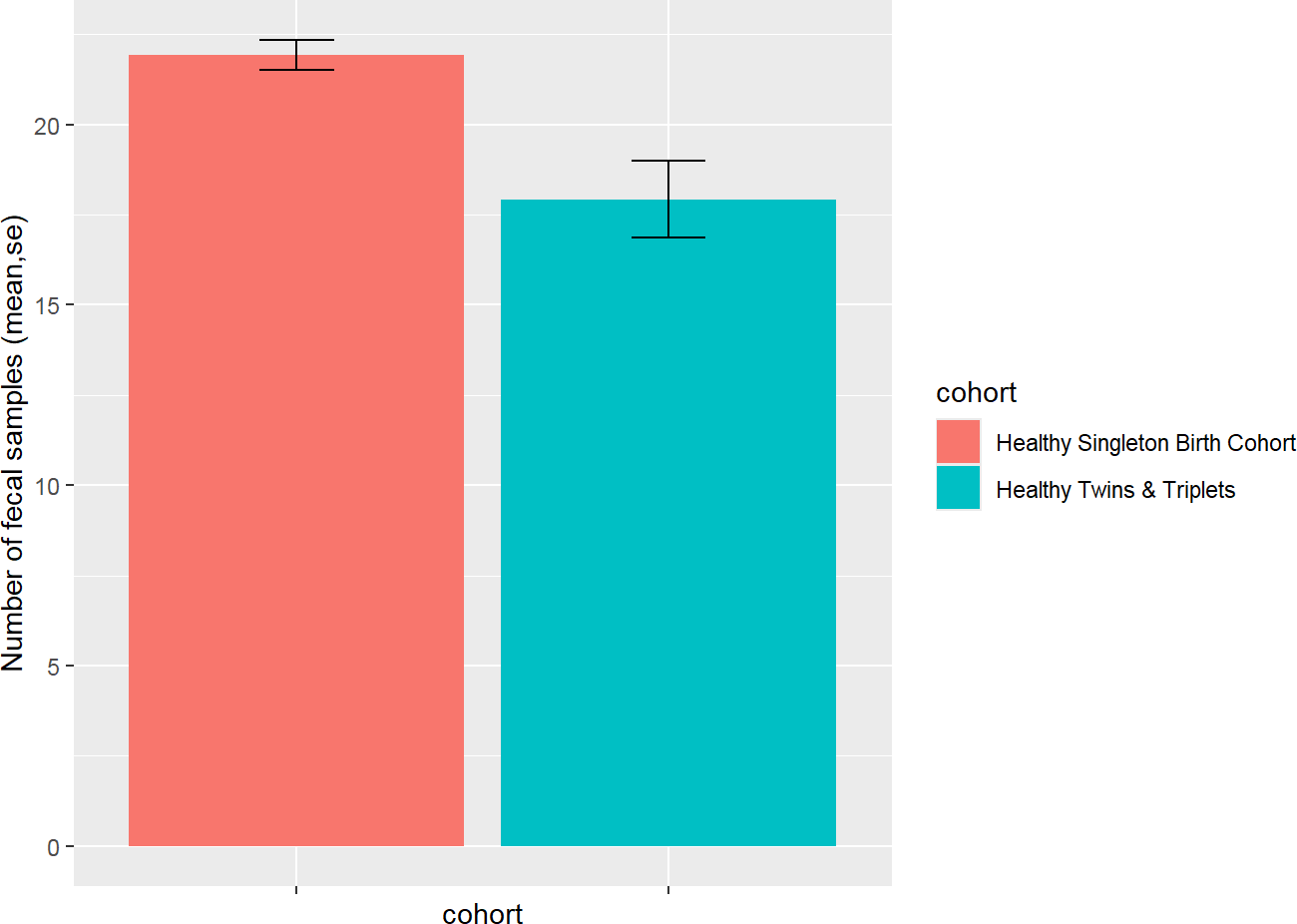
geom\_errorbar(aes(ymin = mean-se, ymax = mean+se),

width = .2, *# Width of the error bars*

position = position\_dodge(.9))+

ylab("Number of fecal samples (mean,se)")+

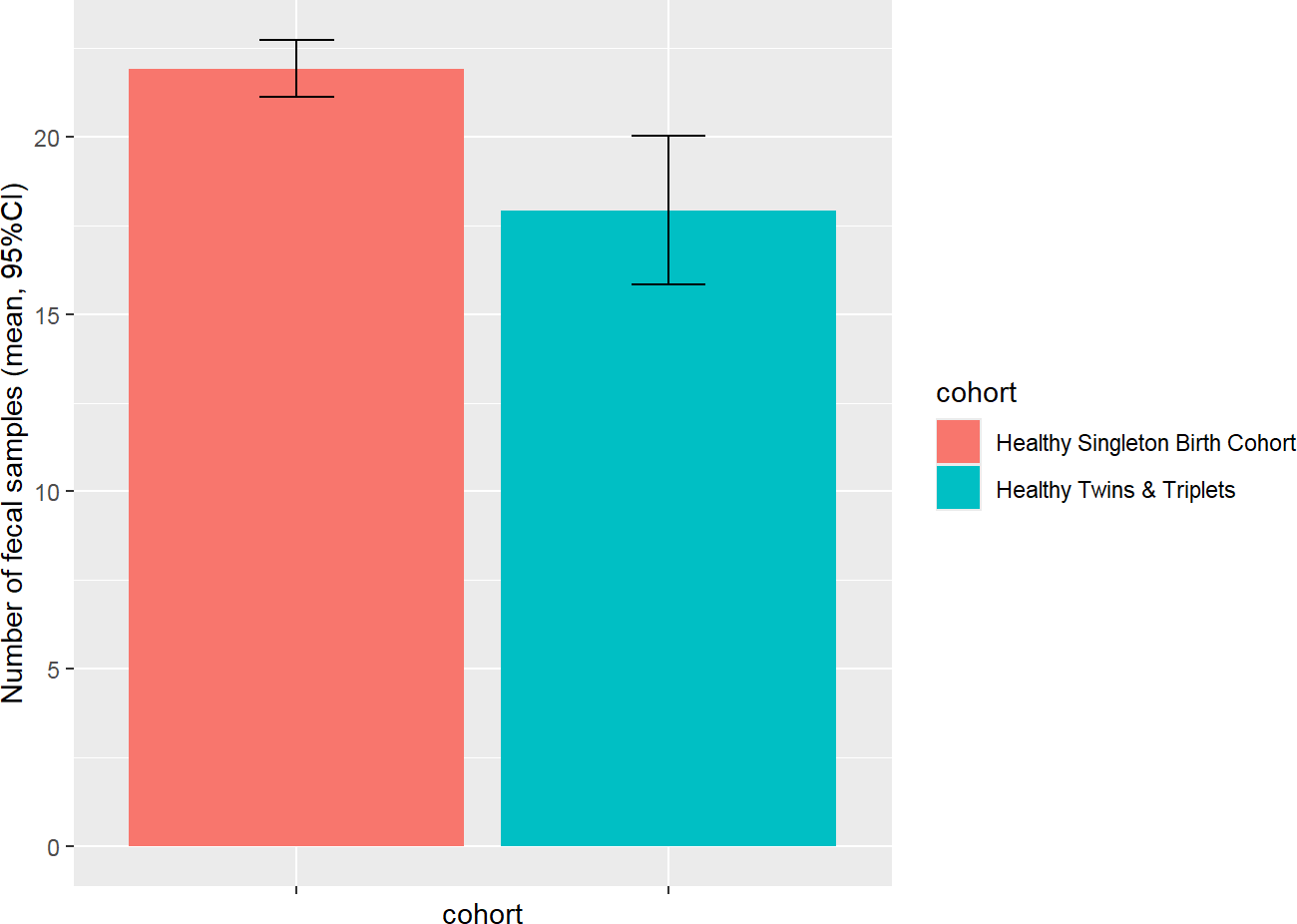
theme(axis.text.x = element\_blank())



Source: Figure by author(s).

**Figure 19**. Bar plot number of fecal samples.

We can replace error bar with 95% confidence interval (CI) for bar plot:



ggplot(nsam, aes(x = cohort, y = mean, fill = cohort)) +

geom\_bar(position = position\_dodge(), stat = "identity") +

geom\_errorbar(aes(ymin = mean-ci, ymax = mean+ci),

width = .2, *# Width of the error bars*

position = position\_dodge(.9))+

ylab("Number of fecal samples (mean, 95%CI)")+

theme(axis.text.x = element\_blank())

Source: Figure by author(s).

**Figure 20**. Bar plot with 95%CI.

We can create a histogram with density for number of fecal samples by cohort of singletons vs twins/triplets:

sh2 %>% group\_by(cohort,child.id) %>% summarise(no.sam = n()) %>%

ggplot(aes(no.sam,y = ..density..,fill = cohort))+

geom\_histogram(bins = 10,alpha = 0.5, position = "identity")+

geom\_density(aes(no.sam,color = cohort),alpha = 0.5)



Source: Figure by author(s).

**Figure 21**. Histogram by cohort.

### Further Exploration

We can do further exploration beyond the practice requirements above. For example, we can do some basic statistical tests.

nsam<-sh2 %>%

group\_by(cohort,child.id) %>%

summarise(no.sam = n())

head(nsam)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | # | A tibble: 6 x 3 |  |  | | |
| ## | # | Groups: cohort | [1] |
| ## |  | cohort |  |  | child.id | no.sam |
| ## |  | <chr> |  |  | <chr> | <int> |
| ## | 1 | Healthy Singleton | Birth | Cohort | Bgsng7018 | 21 |
| ## | 2 | Healthy Singleton | Birth | Cohort | Bgsng7035 | 23 |
| ## | 3 | Healthy Singleton | Birth | Cohort | Bgsng7052 | 20 |
| ## | 4 | Healthy Singleton | Birth | Cohort | Bgsng7063 | 20 |
| ## | 5 | Healthy Singleton | Birth | Cohort | Bgsng7071 | 18 |
| ## | 6 | Healthy Singleton | Birth | Cohort | Bgsng7082 | 18 |

wilcox.test(no.sam~cohort,nsam) ##

## Wilcoxon rank sum test with continuity correction ##

## data: no.sam by cohort

## W = 443.5, p-value = 0.01092

## alternative hypothesis: true location shift is not equal to 0

kruskal.test(no.sam~cohort,nsam)

##

## Kruskal-Wallis rank sum test

##

## data: no.sam by cohort

## Kruskal-Wallis chi-squared = 6.5277, df = 1, p-value = ## 0.01062

t.test(no.sam~cohort,nsam)

##

## Welch Two Sample t-test

##

## data: no.sam by cohort

## t = 3.5045, df = 30.73, p-value = 0.001427

## alternative hypothesis: true difference in means between group Healthy Singleton Birth Cohort and group Healthy Twins & Triplets is not equal to 0

## 95 percent confidence interval:

## 1.671264 6.328736

## sample estimates:

## mean in group Healthy Singleton Birth Cohort

## 21.92

## mean in group Healthy Twins & Triplets

## 17.92

To decide which test is appropriate, the readers may see Chapter “Basic Statistics” of this book.

Note that, there is a difference in data type between sheet 1 and sheet 2:

* Sheet 1 is cross sectional data.
* Sheet 2 is longitudinal data: many observations per “child.id”.

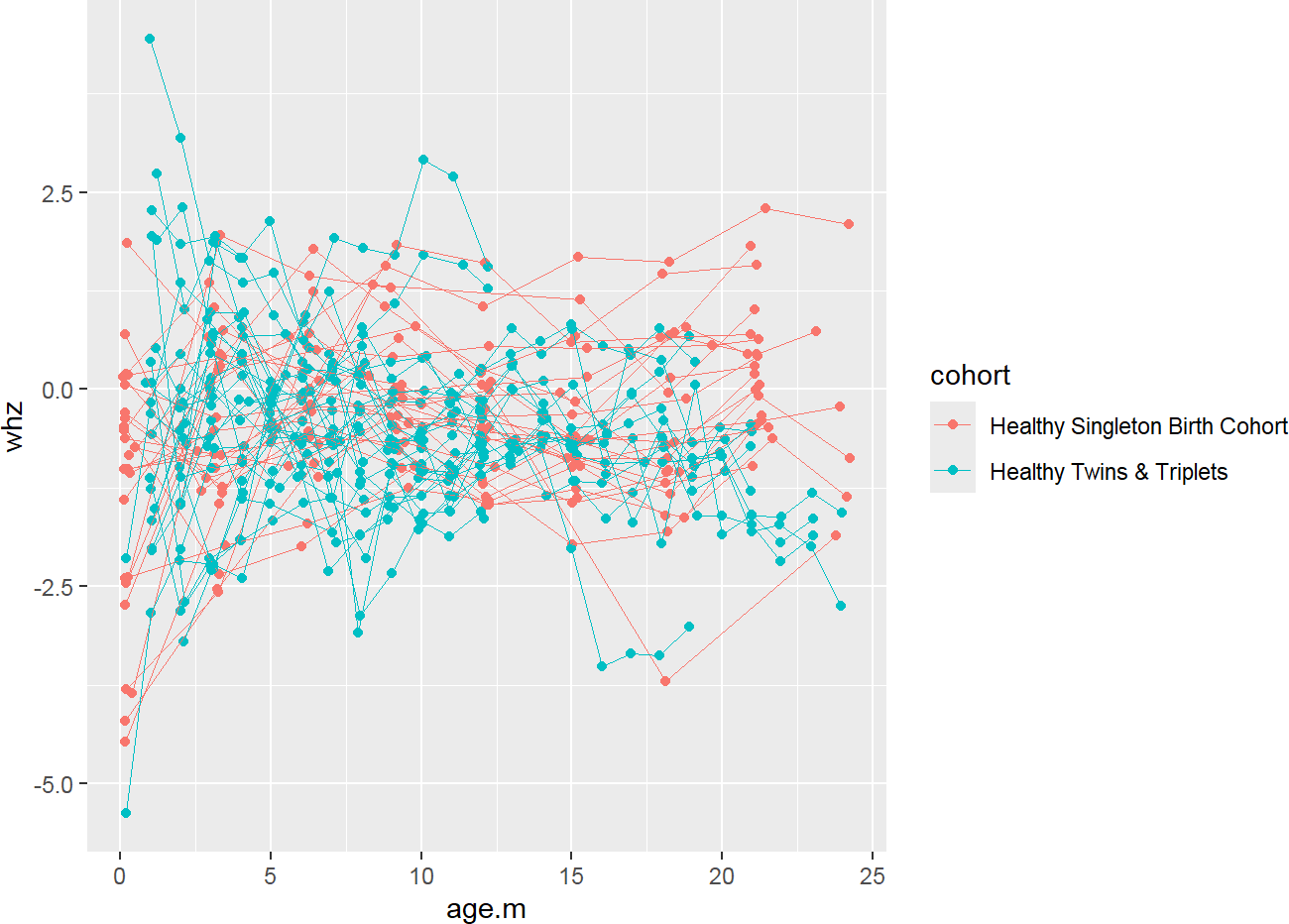
Therefore, different ways of data description and analysis may be used.

For example, for longitudinal data, we can do visualization with a spaghetti plot for “whz”:

sh2 %>% filter(!is.na(whz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = whz, group = child.id, colour = cohort))+

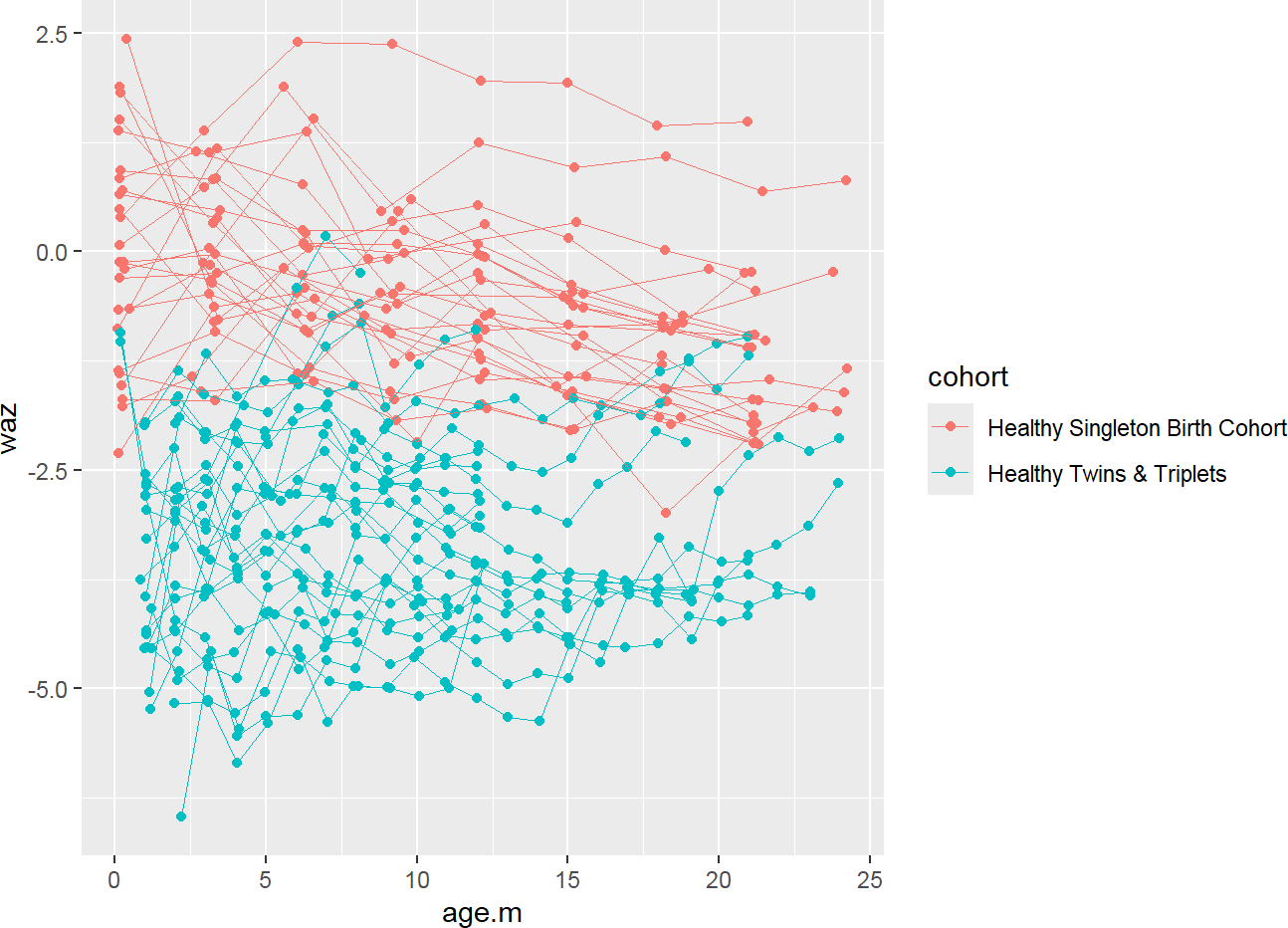
geom\_line(aes(x = age.m, y = whz, group = child.id, colour = cohort),size = 0.3)



Source: Figure by author(s).

**Figure 22**. whz spaghetti plot.

We can do more exploration with “waz”:



sh2 %>% filter(!is.na(waz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = waz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = waz, group = child.id, colour = cohort),size = 0.3)

Source: Figure by author(s).

**Figure 23**. waz spaghetti plot.

We can add fitted lines to the plot:

sh2 %>% filter(!is.na(waz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = waz, group = child.id,

colour = cohort))+

geom\_line(aes(x = age.m, y = waz, group = child.id, colour = cohort), size = 0.3)+

stat\_smooth(aes(x = age.m, y = waz, colour = cohort), method = 'glm', size = 2)



Source: Figure by author(s).

**Figure 24**. waz spaghetti plot with fitted line.

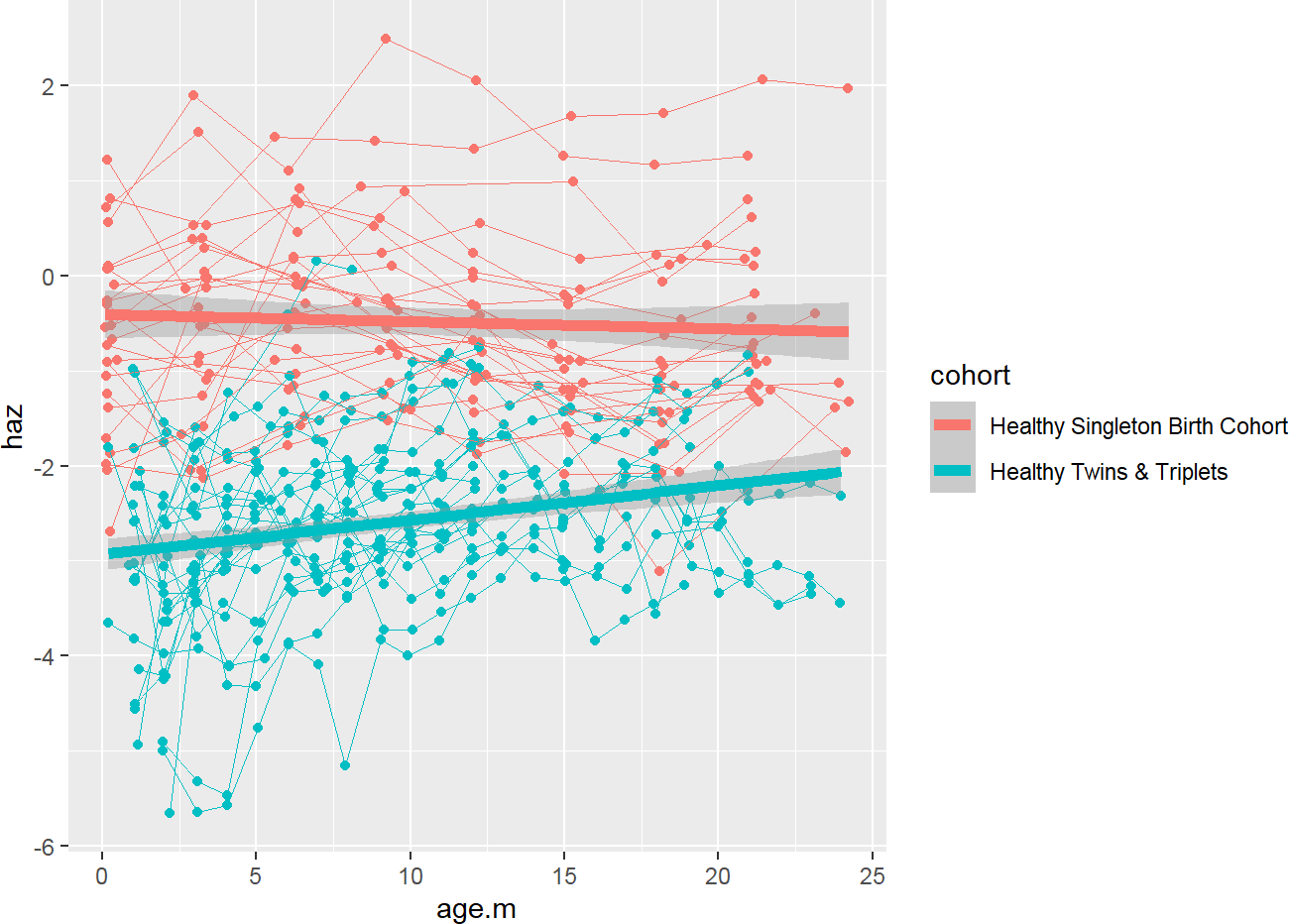
We can do more similar exploration with “haz”:

sh2 %>% filter(!is.na(haz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = haz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = haz, group = child.id, colour = cohort), size = 0.3)+

stat\_smooth(aes(x = age.m, y = haz, colour = cohort), method = 'glm', size = 2)



Source: Figure by author(s).

**Figure 25**. haz spaghetti plot with fitted line.

To test for the difference between two groups, please check out the chapter “Longitudinal data analysis with R” in this book.

We can combine all the above plots into an all-in-one spaghetti plot of three indices by cohort adding each item to the plot and using *facet\_grid()*:

sh2 %>%

filter(!is.na(haz) & !is.na(waz) & !is.na(whz)) %>% ggplot()+

*#add each item to the plot*

geom\_point(aes(x = age.m, y = haz,

group = child.id,color = "haz"))+ geom\_line(aes(x = age.m, y = haz,

group = child.id,color = "haz"), size = 0.3)+

stat\_smooth(aes(x = age.m, y = haz,

color = "haz"), method = 'glm',size = 2)+ geom\_point(aes(x = age.m, y = waz,

group = child.id,color = "waz"))+ geom\_line(aes(x = age.m, y = waz,

group = child.id,color = "waz"), size = 0.3)+

stat\_smooth(aes(x = age.m, y = waz,

color = "waz"), method = 'glm',size = 2)+ geom\_point(aes(x = age.m, y = whz,

group = child.id,color = "whz"))+ geom\_line(aes(x = age.m, y = whz,

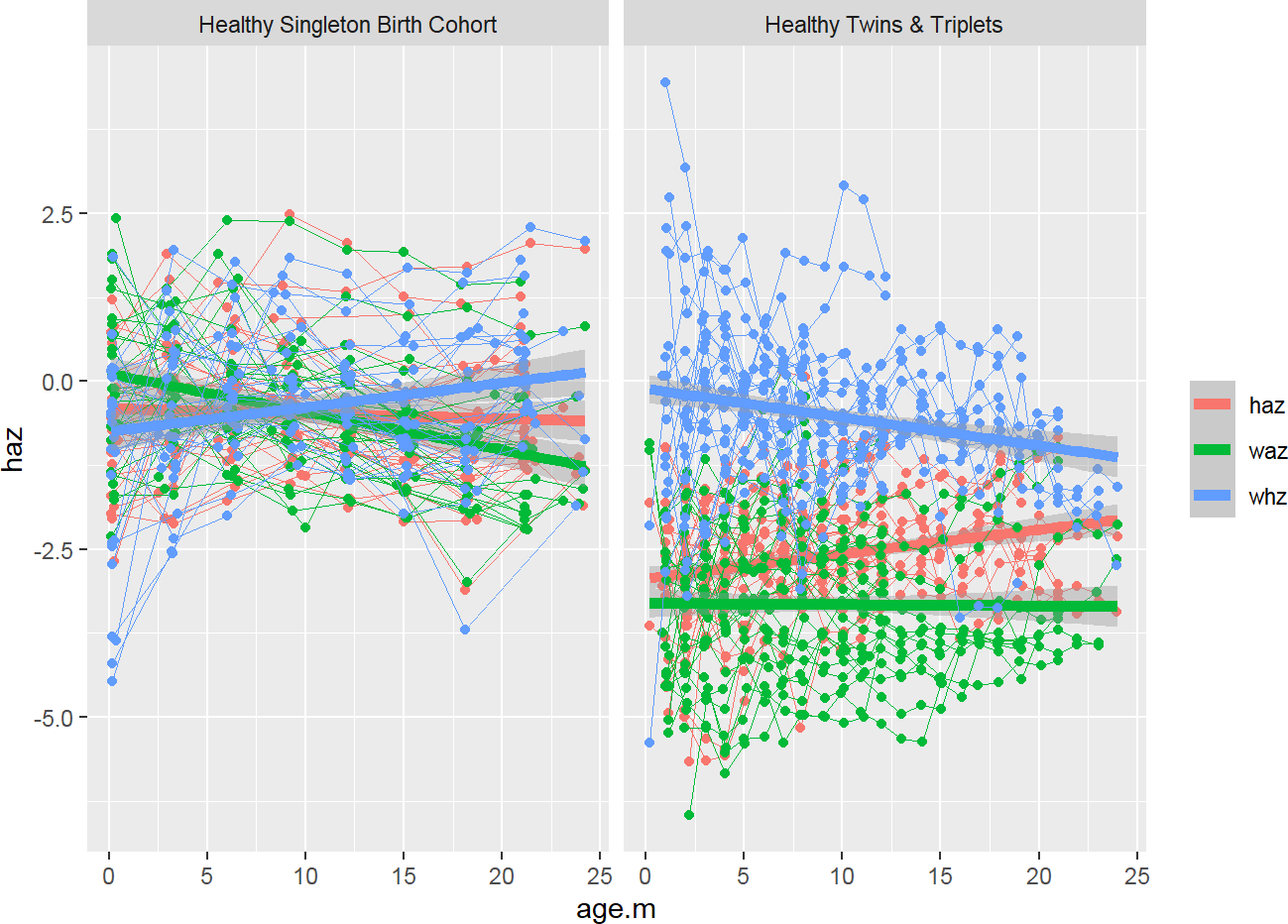
group = child.id,color = "whz"), size = 0.3)+

stat\_smooth(aes(x = age.m, y = whz,

color = "whz"), method = 'glm', size = 2)+

*#divide the plot into multiple panels (cohorts)* facet\_grid(.~ cohort)+

theme(legend.title = element\_blank())



Source: Figure by author(s).

**Figure 26**. All-in-one spaghetti plot.

We can create each plot for each index by cohort and then combine all plots into an all-in-one spaghetti plot using *ggarrange()* of the “*ggpubr*” package [20]:

p1<-sh2 %>%

filter(!is.na(haz) & !is.na(waz) & !is.na(whz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = haz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = haz, group = child.id, colour = cohort), size = 0.3)+

stat\_smooth(aes(x = age.m, y = haz, colour = cohort), method = 'glm', size = 2)+

theme(legend.position = "bottom")

p2<-sh2 %>%

filter(!is.na(haz) & !is.na(waz) & !is.na(whz)) %>%

ggplot()+

geom\_point(aes(x = age.m, y = waz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = waz, group = child.id, colour = cohort), size = 0.3)+

stat\_smooth(aes(x = age.m, y = waz, colour = cohort), method = 'glm', size = 2)+

theme(legend.position = "bottom")

p3<-sh2 %>%

filter(!is.na(haz) & !is.na(waz) & !is.na(whz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = whz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = whz, group = child.id, colour = cohort), size = 0.3)+

stat\_smooth(aes(x = age.m, y = whz, colour = cohort), method = 'glm',size = 2)+

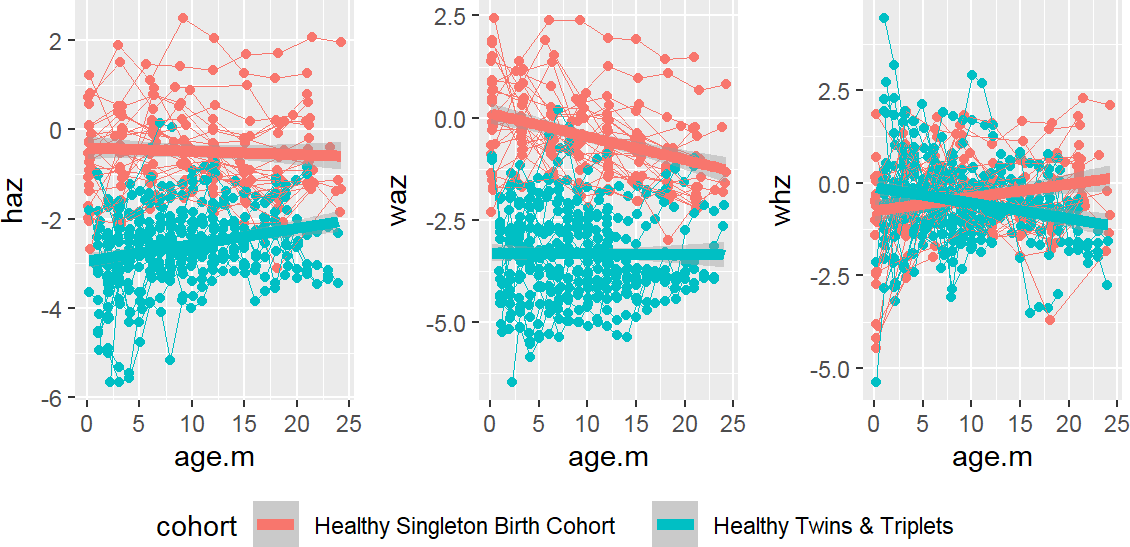
theme(legend.position = "bottom")

*# Combine 3 plots into 1 plot*

*#install.packages("ggpubr")*

library(ggpubr)

ggarrange(p1, p2, p3,nrow = 1, common.legend = TRUE, legend= "bottom")



Source: Figure by author(s).

**Figure 27**. All-in-one spaghetti plot for each index.

## Useful Resources

### R Books

* + - * [Population Health Data Science with R](https://bookdown.org/medepi/phds/) [21].
      * [Modern R with the tidyverse](https://b-rodrigues.github.io/modern_R/index.html) [22].

### Datasets

* + - * The datasets used in some examples of this book are of Bangladeshi infants published by [Subramanian et al](https://www.nature.com/articles/nature13421) [13]. These datasets may be downloaded from [Gordon’s lab website](https://gordonlab.wustl.edu/), or from my [github repo](https://github.com/nhanhocu/metamicrobiomeR/tree/master/inst/extdata/QIIME_outputs/Bangladesh) (https://github.com/nhanhocu/metamicrobiomeR/tree/master/inst/extdata/QIIME\_outputs/Bangladesh).

## Chapter Summary

* Data Import: the “*rio”* package, particularly the *import()* and *import\_list()*

functions are highly useful.

* Data manipulation: the “*tidyverse”* collection of packages, especially the key functions from the “*dplyr”* package like *filter(),* *select()*, *rename()*, *mutate()* are very handy.
* Data Visualization: the “*ggplot2”* package is the primary tool for creating a wide

range of graphics.

* Descriptive statistics: the *summarise()* function from “*dplyr”*, often in combination with *group\_by()* for group-level summaries is handy.
* Creating summary tables: the “*arsenal”* package (with function *tableby()*) is among useful packages.

# Basic Statistics with R

This chapter introduces foundational statistical concepts and their practical implementation using R. It is essential to first understand the nature of biomedical data, particularly its distribution, as this determines the choice of appropriate statistical tests. This chapter covers the two most critical data distributions (the Normal Distribution for continuous data and the Binomial Distribution for binary outcomes) and provides the essential R functions for working with them. Furthermore, this chapter introduces the readers the most commonly used statistical tests (the t-test and its non-parametric alternatives (Wilcoxon and Kruskal-Wallis tests) for comparison of continuous variables, and the Chi-square/Fisher's Exact tests for comparisons of categorical variables). This chapter also includes a section about power-sample size estimation for bio-medical studies.

## Data Distribution

Understanding the distribution of the data would help with choosing proper statistical tests and interpret the findings. The two most common and important distribution (normal and binomial distribution) will be discussed.

First, we load the main R packages used in this chapter (*"knitr" [5], "rmarkdown" [6], "kableExtra" [12], "arsenal" [19], "car" [23], "pwr" [24], "samplesizeestimator" [25], "presize"* [26]):

*#load multiple packages*

Packages <- c("knitr", "rmarkdown", "kableExtra", "arsenal", "car", "pwr", "samplesizeestimator", "presize")

lapply(Packages, library, character.only = TRUE)

### The Normal Distribution

According to the Central Limit Theorem, the distribution of sample means from any population will tend to be normal as the sample size increases. The normal distribution is defined by two key parameters: the mean (*µ*), which marks its center, and the standard deviation (*σ*), which describes its spread [27]. Below are key R functions related to normal distribution:

* + - * *dnorm(x, mean, sd)*: Density. Calculates the height of the curve at a point x.
      * *pnorm(q, mean, sd)*: Probability. Calculates the cumulative probability (area under the curve) up to a point q.
      * *qnorm(p, mean, sd)*: Quantile. The inverse of *pnorm*. Finds the value x associated with a cumulative probability p.
      * *rnorm(n, mean, sd)*: Random. Generates n random numbers from a normal distribution.

Below is an example to visualize the normal distribution of adult male height in a population with a mean of 178 cm and a standard deviation of 7 cm.

*# First, generate a large sample of random data to simulate the population*

set.seed(123) *# for reproducibility*

simulated\_heights <- rnorm(1000, mean = 178, sd = 7) *#Create a histogram to see the sample's distribution* hist(simulated\_heights,

breaks = 30, *# Suggest more bins for a smoother look*

freq = FALSE, *# Plot density, not frequency, to allow overlay*

main = "Distribution of Adult Male Heights", xlab = "Height (cm)",

col = "lightblue")

*#Overlay the theoretical normal curve*

curve(dnorm(x, mean = 178, sd = 7), from = 150, to = 210,

add = TRUE, *# Add to the existing plot*

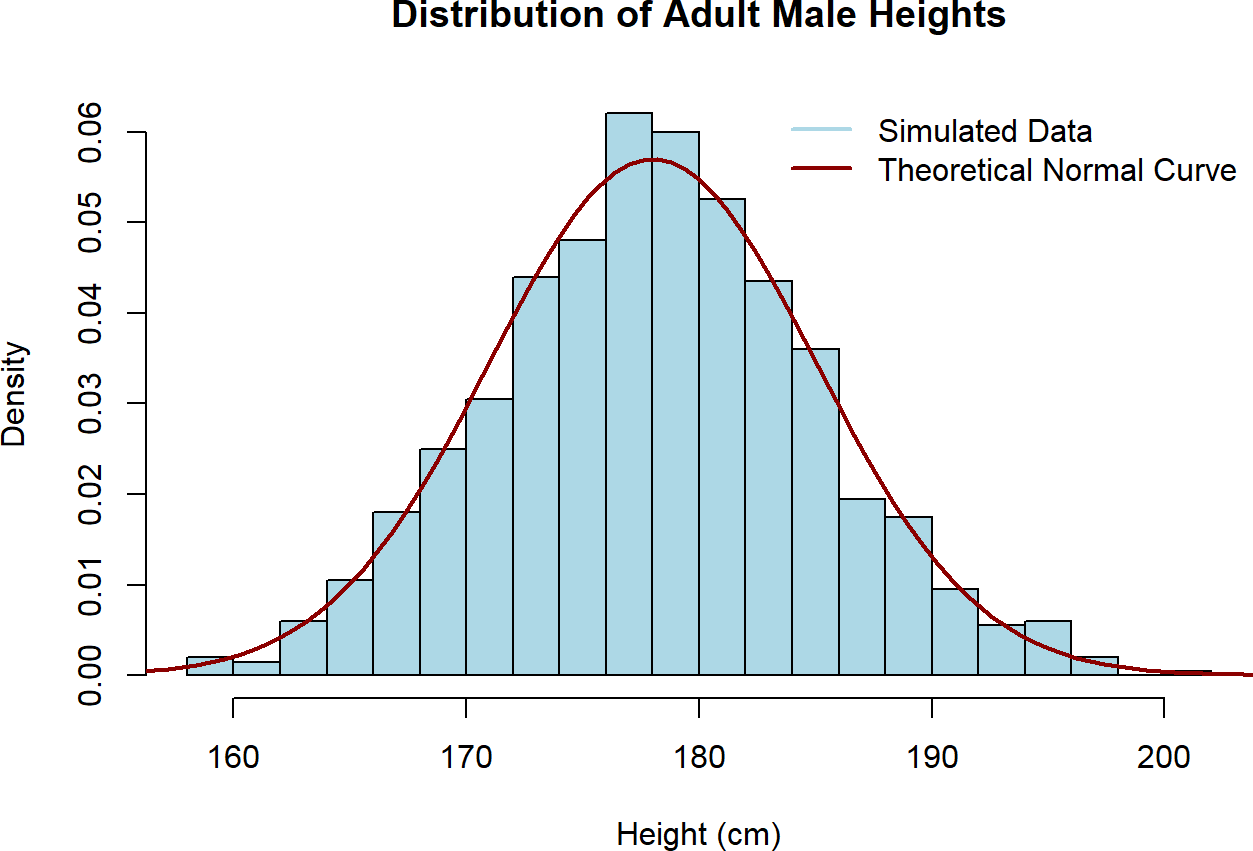
col = "darkred",

lwd = 2) *# Line width #Add a legend*

legend("topright", legend = c("Simulated Data",

"Theoretical Normal Curve"),

col = c("lightblue", "darkred"), lty = 1, lwd = 2, bty = "n")



Source: Figure by author(s).

**Figure 28**. Example normal distribution.

### The Binomial Distribution

The binomial distribution is used for discrete data. It models the number of “successes” in a fixed number (n or size) of independent trials, where the probability of success (p) is the same for each trial [28]. For example: In a trial of 20 patients, a new drug has a known response probability of 40%, what is the probability that exactly 8 patients will respond? We use *dbinom()* (the density or, for discrete data, the probability mass function).

dbinom(x = 8, size = 20, prob = 0.40)

## [1] 0.1797058

The output shows that there is approximately an 18.0% chance that exactly 8 out of 20 patients will respond. Another question is: what is the probability that “8 or fewer” patients will respond? We use *pbinom()* for the cumulative probability.

pbinom(q = 8, size = 20, prob = 0.40)

## [1] 0.5955987

The output shows that there is a 59.6% chance that 8 or fewer patients will respond to the treatment.

We can visualize the entire probability distribution for this example.

*# Possible outcomes: 0 to 20 successes*

outcomes <- 0:20

*# Calculate the probability for each outcome*

probabilities <- dbinom(x = outcomes, size = 20, prob = 0.40)

*# Create a bar plot*

barplot(probabilities,

names.arg = outcomes,

main = "Binomial Distribution (n = 20, p = 0.4)",

xlab = "Number of Responding Patients",

ylab = "Probability",

col = "skyblue")



Source: Figure by author(s).

**Figure 29**. Example binomial distribution.

The bar plot shows the probability of each possible outcome, from 0 successes to 20. We can see the most likely outcome is 8 successes, and the probabilities tail off as we move away from the mean (which is np = 20\*0.4 = 8).

The relationship between binomial distribution and normal distribution:

For large n, the binomial distribution approximates the normal distribution with mean = np and variance = np(1 – p). This approximation explains why proportions can often be analyzed using normal-based methods when sample size is large.

## Basic Statistical Tests

### Tests for Comparing Continuous Variables

Continuous variables (e.g., BMI, cholesterol, blood pressure) are compared between groups using parametric tests when the data are approximately normal and non-parametric tests otherwise.

#### Compare mean to a constant

To test if the mean of a single sample significantly differs from a known or hypothesized population mean. T-test may be used in this case with assumption:

* + - * Independence of observations.
      * Data are approximately normally distributed.

Example: Using the built-in iris dataset, we test if the mean sepal length (“Sepal.Length”) of the virginica species is significantly different from a hypothesized value of 6.1 cm. First, we check for normal distribution with Q-Q plot using *qqnorm()* and *qqline()* then perform t-test.

*# get the data*

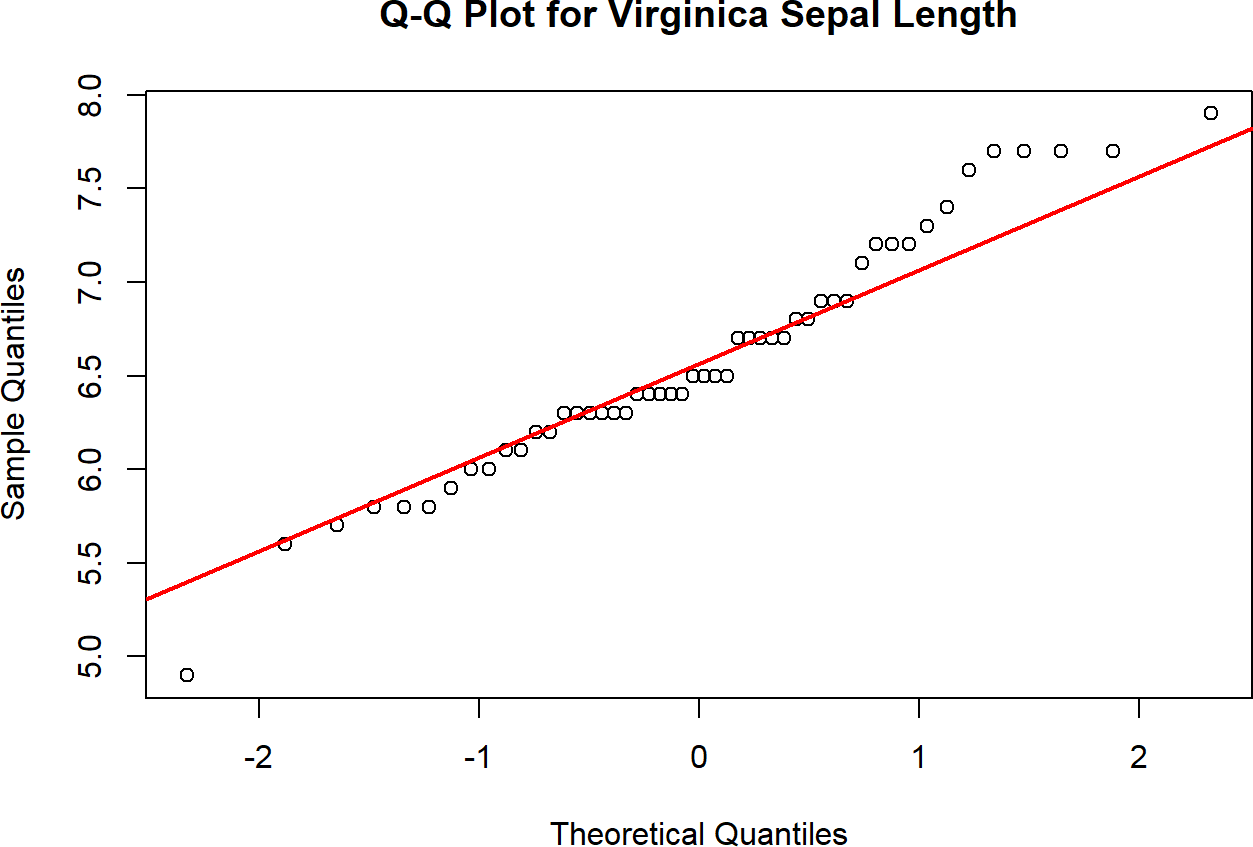
virginica\_sepal\_length <- iris$Sepal.Length[iris$Species == "virginica"]

*# Check the normality assumption with a Q-Q Plot*

qqnorm(virginica\_sepal\_length,

main = "Q-Q Plot for Virginica Sepal Length")

qqline(virginica\_sepal\_length, col = "red", lwd = 2)



Source: Figure by author(s).

**Figure 30**. Q-Q Plot.

*# 3. Perform the t-test*

t.test(virginica\_sepal\_length, mu = 6.1)

##

## One Sample t-test

##

## data: virginica\_sepal\_length

## t = 5.4266, df = 49, p-value = 0.000001765

## alternative hypothesis: true mean is not equal to 6.1

## 95 percent confidence interval:

## 6.407285 6.768715

## sample estimates:

## mean of x

## 6.588

The data falls roughly along the straight line (approximates normal distribution) so t-test may be used. The results show that the mean is significantly different from 6.1 (p-value <0.05 and 95%CI confidence interval does not include 6.1).

#### Compare means of two independent groups

The assumptions are:

* + - * Independence of the two groups.
      * Approximate normality within each group.
      * Homogeneity of variances (the variances in the two groups are roughly equal). R’s default *t.test()* uses the Welch t-test, which does not require this assumption and is therefore a robust choice. Levene’s test may be used to test for this assumption.

The below example will test if there is a significant difference in the miles per gallon (mpg) between cars with automatic (am = 0) and manual (am = 1) transmissions in the *mtcars* dataset of the “*car*” package [23]. Check for homogeneity of variances using Levene’s test (optional but good practice):

*#library(car)*

leveneTest(mpg ~ as.factor(am), data = mtcars)

## Levene's Test for Homogeneity of Variance (center = median) ## Df F value Pr(>F)

## group 1 4.1876 0.04957 \*

## 30

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The p-value is > 0.05, so we do not reject the null of equal variances. Even so, the Welch t-test is a safe default.

Now we perform the t-test:

t.test(mpg ~ am, data = mtcars) ##

## Welch Two Sample t-test

##

## data: mpg by am

## t = -3.7671, df = 18.332, p-value = 0.001374

## alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0

## 95 percent confidence interval:

## -11.280194 -3.209684

## sample estimates:

## mean in group 0 mean in group 1

## 17.14737 24.39231

The results show that the two sample means are significantly different (p-value

<0.05, 95%CI of the difference does not include 0).

#### Non-parametric alternative to t-test

If normal distribution assumption does not meet, Wilcoxon’s rank sum test (*wilcox.test()*) may be used instead of t-test to compare continuous variables of two groups.

wilcox.test(mpg ~ am, data = mtcars)

##

## Wilcoxon rank sum test with continuity correction

##

## data: mpg by am

## W = 42, p-value = 0.001871

## alternative hypothesis: true location shift is not equal to 0

This test compares medians rather than means and does not require normality. Interpretation: a significant result means the distributions differ in central tendency between group. The results should be reported with median and interquartile range (IQR) for each group instead of mean (SD).

For comparing between two or more groups, Kruskal Wallis’ test (*kruskal.test()*) may be used.

kruskal.test(mpg ~ am, data = mtcars) ##

## Kruskal-Wallis rank sum test

##

## data: mpg by am

## Kruskal-Wallis chi-squared = 9.7914, df = 1, p-value = ## 0.001753

### Tests for Comparing Categorical Variables

Chi-square test (*chisq.test()*) may be used to test for a statistically significant association between two categorical variables with assumption:

* Data are counts,
* Independence of observations,
* Adequate expected cell counts (expected counts > 5).

The following example will test if there is an association between the number of gears (“gear”) and the number of carburetors (“carb”) using the *mtcars* dataset.

*# Create a contingency table*

contingency\_table <- table(mtcars$gear, mtcars$carb)

contingency\_table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## |  | | | | | | |
| ## |  | 1 | 2 | 3 | 4 | 6 | 8 |
| ## | 3 | 3 | 4 | 3 | 5 | 0 | 0 |
| ## | 4 | 4 | 4 | 0 | 4 | 0 | 0 |
| ## | 5 | 0 | 2 | 0 | 1 | 1 | 1 |

*# Perform the test*

chisq\_result <- chisq.test(contingency\_table)

chisq\_result

##

## Pearson's Chi-squared test

##

## data: contingency\_table

## X-squared = 16.518, df = 10, p-value = 0.08573

*# Check the expected counts assumption*

chisq\_result$expected

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## |  | | | | | | |
| ## |  | 1 | 2 | 3 | 4 | 6 | 8 |
| ## | 3 | 3.28125 | 4.6875 | 1.40625 | 4.6875 | 0.46875 | 0.46875 |
| ## | 4 | 2.62500 | 3.7500 | 1.12500 | 3.7500 | 0.37500 | 0.37500 |
| ## | 5 | 1.09375 | 1.5625 | 0.46875 | 1.5625 | 0.15625 | 0.15625 |

For small samples or when expected cell counts < 5, use Fisher’s Exact Test (*fisher.test()*). Fisher’s test computes the exact probability of the observed configuration (and more extreme ones) under the null hypothesis of independence. It is reliable even for very small datasets which are commonly used in pilot studies or rare-event analyses. In the example above, m

fisher.test(contingency\_table)

##

## Fisher's Exact Test for Count Data

##

## data: contingency\_table

## p-value = 0.2434

## alternative hypothesis: two.sided

## Data Summary and Basic Statistical Test in a Table

As introduced in the previous chapter, the “*arsenal”* package may be used to summarize and perform corresponding statistical tests for many variables in a dataset [19]. The below example summarizes and compares all the variables in the dataset *mtcars* between *am* groups and presents the results in a publication ready table:

*#library(arsenal)*

varsum<-colnames(mtcars)[!colnames(mtcars) %in% "am"] mylabels <-as.list(varsum)

names(mylabels)<-varsum

tab1 <- tableby(as.formula(paste("am",paste(varsum, collapse = "+"),

sep = "~")),

data = mtcars)

kable(summary(tab1,

labelTranslations = mylabels, text = TRUE),

longtable = TRUE,

caption = "Summary, statistical tests for all variables")%>% kable\_styling(latex\_options = c("scale\_down",

"hold\_position",

"repeat\_header"))

**Table 5**. Summary, statistical tests for all variables.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0 (N = 19)** | **1 (N = 13)** | **Total (N = 32)** | **p Value** |
| mpg |  |  |  | < 0.001 |
| - Mean (SD) | 17.147 (3.834) | 24.392 (6.167) | 20.091 (6.027) |  |
| - Range | 10.400−24.400 | 15.000−33.900 | 10.400−33.900 |  |
| cyl |  |  |  | 0.002 |
| - Mean (SD) | 6.947 (1.545) | 5.077 (1.553) | 6.188 (1.786) |  |
| - Range | 4.000−8.000 | 4.000−8.000 | 4.000−8.000 |  |
| disp |  |  |  | < 0.001 |
| - Mean (SD) | 290.379 (110.172) | 143.531 (87.204) | 230.722 (123.939) |  |
| - Range | 120.100−472.000 | 71.100−351.000 | 71.100−472.000 |  |
| hp |  |  |  | 0.180 |
| - Mean (SD) | 160.263 (53.908) | 126.846 (84.062) | 146.688 (68.563) |  |
| - Range | 62.000−245.000 | 52.000−335.000 | 52.000−335.000 |  |
| drat |  |  |  | < 0.001 |
| - Mean (SD) | 3.286 (0.392) | 4.050 (0.364) | 3.597 (0.535) |  |
| - Range | 2.760−3.920 | 3.540−4.930 | 2.760−4.930 |  |
| wt |  |  |  | < 0.001 |
| - Mean (SD) | 3.769 (0.777) | 2.411 (0.617) | 3.217 (0.978) |  |
| - Range | 2.465−5.424 | 1.513−3.570 | 1.513−5.424 |  |
| qsec |  |  |  | 0.206 |
| - Mean (SD) | 18.183 (1.751) | 17.360 (1.792) | 17.849 (1.787) |  |
| - Range | 15.410−22.900 | 14.500−19.900 | 14.500−22.900 |  |
| vs |  |  |  | 0.357 |
| - Mean (SD) | 0.368 (0.496) | 0.538 (0.519) | 0.438 (0.504) |  |
| - Range | 0.000−1.000 | 0.000−1.000 | 0.000−1.000 |  |
| gear |  |  |  | < 0.001 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Table 5.** *Cont.* |  | |
|  | **0 (N = 19)** | **1 (N = 13)** | **Total (N = 32)** | **p Value** |
| - Mean (SD) | 3.211 (0.419) | 4.385 (0.506) | 3.688 (0.738) |  |
| - Range | 3.000−4.000 | 4.000−5.000 | 3.000−5.000 |  |
| carb |  |  |  | 0.754 |
| - Mean (SD) | 2.737 (1.147) | 2.923 (2.178) | 2.812 (1.615) |  |
| - Range | 1.000−4.000 | 1.000−8.000 | 1.000−8.000 |  |

Source: Table by author(s).

The summary options and statistical tests by the *tableby()* may be adjusted. Below is an example for setting the options with *tableby.control()* then re-applying the *tableby()* function:

*# set summary options and statistical tests of choice*

my\_controls <- tableby.control(

test = T,

total = T,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = c("meansd", "medianq1q3","meanCI","range","Nmiss2"),

cat.stats = c("countpct", "Nmiss2"),

stats.labels = list(

meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)",

range = "Min - Max",

Nmiss2 = "Missing",

meanCI = "Mean (95%CI)"

)

)

*# Apply the tableby function again:*

varsum<-colnames(mtcars)[!colnames(mtcars) %in% "am"]

mylabels <-as.list(varsum)

names(mylabels)<-varsum

tab1 <- tableby(as.formula(paste("am", paste(varsum,collapse = "+"),

sep = "~")),

data = mtcars,

control = my\_controls) kable(summary(tab1,

labelTranslations = mylabels, text = TRUE),

caption = "Summary, statistical tests, adjusted options")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 6**. Summary, statistical tests, adjusted options.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0 (N = 19)** | **1 (N = 13)** | **Total (N = 32)** | **p Value** |
| mpg |  |  |  | 0.002 |
| - Mean (SD) | 17.147 (3.834) | 24.392 (6.167) | 20.091 (6.027) |  |
| - Median (Q1, Q3) | 17.300 (14.950, 19.200) | 22.800 (21.000, 30.400) | 19.200 (15.425, 22.800) |  |
| - Mean (95%CI) | 17.147 (15.299, 18.995) | 24.392 (20.666, 28.119) | 20.091 (17.918, 22.264) |  |
| - Min−Max | 10.400−24.400 | 15.000−33.900 | 10.400−33.900 |  |
| - Missing | 0 | 0 | 0 |  |
| cyl |  |  |  | 0.004 |
| - Mean (SD) | 6.947 (1.545) | 5.077 (1.553) | 6.188 (1.786) |  |
| - Median (Q1, Q3) | 8.000 (6.000, 8.000) | 4.000 (4.000, 6.000) | 6.000 (4.000, 8.000) |  |
| - Mean (95%CI) | 6.947 (6.203, 7.692) | 5.077 (4.139, 6.015) | 6.188 (5.544, 6.831) |  |
| - Min−Max | 4.000−8.000 | 4.000−8.000 | 4.000−8.000 |  |
| - Missing | 0 | 0 | 0 |  |
| disp |  |  |  | < 0.001 |
| - Mean (SD) | 290.379 (110.172) | 143.531 (87.204) | 230.722 (123.939) |  |
| - Median (Q1, Q3) | 275.800 (196.300, 360.000) | 120.300 (79.000, 160.000) | 196.300 (120.825, 326.000) |  |
| - Mean (95%CI) | 290.379 (237.278, 343.480) | 143.531 (90.834, 196.228) | 230.722 (186.037, 275.407) |  |
| - Min−Max | 120.100−472.000 | 71.100−351.000 | 71.100−472.000 |  |
| - Missing | 0 | 0 | 0 |  |
| hp |  |  |  | 0.044 |
| - Mean (SD) | 160.263 (53.908) | 126.846 (84.062) | 146.688 (68.563) |  |
| - Median (Q1, Q3) | 175.000 (116.500, 192.500) | 109.000 (66.000, 113.000) | 123.000 (96.500, 180.000) |  |
| - Mean (95%CI) | 160.263 (134.280, 186.246) | 126.846 (76.048, 177.645) | 146.688 (121.968, 171.407) |  |
| - Min−Max | 62.000−245.000 | 52.000−335.000 | 52.000−335.000 |  |
| - Missing | 0 | 0 | 0 |  |
| drat |  |  |  | < 0.001 |
| - Mean (SD) | 3.286 (0.392) | 4.050 (0.364) | 3.597 (0.535) |  |
| - Median (Q1, Q3) | 3.150 (3.070, 3.695) | 4.080 (3.850, 4.220) | 3.695 (3.080, 3.920) |  |
| - Mean (95%CI) | 3.286 (3.097, 3.475) | 4.050 (3.830, 4.270) | 3.597 (3.404, 3.789) |  |
| - Min−Max | 2.760−3.920 | 3.540−4.930 | 2.760−4.930 |  |
| - Missing | 0 | 0 | 0 |  |
| wt |  |  |  | < 0.001 |
| - Mean (SD) | 3.769 (0.777) | 2.411 (0.617) | 3.217 (0.978) |  |
| - Median (Q1, Q3) | 3.520 (3.438, 3.843) | 2.320 (1.935, 2.780) | 3.325 (2.581, 3.610) |  |
| - Mean (95%CI) | 3.769 (3.394, 4.144) | 2.411 (2.038, 2.784) | 3.217 (2.864, 3.570) |  |
| - Min−Max | 2.465−5.424 | 1.513−3.570 | 1.513−5.424 |  |
| - Missing | 0 | 0 | 0 |  |
| qsec |  |  |  | 0.258 |
| - Mean (SD) | 18.183 (1.751) | 17.360 (1.792) | 17.849 (1.787) |  |
| - Median (Q1, Q3) | 17.820 (17.175, 19.170) | 17.020 (16.460, 18.610) | 17.710 (16.892, 18.900) |  |
| - Mean (95%CI) | 18.183 (17.339, 19.027) | 17.360 (16.277, 18.443) | 17.849 (17.204, 18.493) |  |
| - Min−Max | 15.410−22.900 | 14.500−19.900 | 14.500−22.900 |  |
| - Missing | 0 | 0 | 0 |  |

**Table 6.** *Cont.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0 (N = 19)** | **1 (N = 13)** | **Total (N = 32)** | **p Value** |
| vs |  |  |  | 0.349 |
| - Mean (SD) | 0.368 (0.496) | 0.538 (0.519) | 0.438 (0.504) |  |
| - Median (Q1, Q3) | 0.000 (0.000, 1.000) | 1.000 (0.000, 1.000) | 0.000 (0.000, 1.000) |  |
| - Mean (95%CI) | 0.368 (0.130, 0.607) | 0.538 (0.225, 0.852) | 0.438 (0.256, 0.619) |  |
| - Min−Max | 0.000−1.000 | 0.000−1.000 | 0.000−1.000 |  |
| - Missing | 0 | 0 | 0 |  |
| gear |  |  |  | < 0.001 |
| - Mean (SD) | 3.211 (0.419) | 4.385 (0.506) | 3.688 (0.738) |  |
| - Median (Q1, Q3) | 3.000 (3.000, 3.000) | 4.000 (4.000, 5.000) | 4.000 (3.000, 4.000) |  |
| - Mean (95%CI) | 3.211 (3.009, 3.412) | 4.385 (4.079, 4.691) | 3.688 (3.421, 3.954) |  |
| - Min−Max | 3.000−4.000 | 4.000−5.000 | 3.000−5.000 |  |
| - Missing | 0 | 0 | 0 |  |
| carb |  |  |  | 0.720 |
| - Mean (SD) | 2.737 (1.147) | 2.923 (2.178) | 2.812 (1.615) |  |
| - Median (Q1, Q3) | 3.000 (2.000, 4.000) | 2.000 (1.000, 4.000) | 2.000 (2.000, 4.000) |  |
| - Mean (95%CI) | 2.737 (2.184, 3.290) | 2.923 (1.607, 4.239) | 2.812 (2.230, 3.395) |  |
| - Min−Max | 1.000−4.000 | 1.000−8.000 | 1.000−8.000 |  |
| - Missing | 0 | 0 | 0 |  |

Source: Table by author(s).

Readers may try other packages like “*tableone*” [29] or “*gtsummary*” [30] which may also generate publication ready baseline tables for biomedical papers.

## Sample Size—Power Estimation

A well-designed study has a high probability of detecting a true effect. The probability for detecting a true effect of a study is its “statistical power”. A good study should have a power >= 0.8.

Power analysis typically involves the following components [31]:

* Sample Size: the number of subjects in the study.
* Effect Size (e.g., Cohen’s d): The magnitude of the effect (e.g., a large difference

is easier to detect than a small one).

* Significance Level: the threshold for statistical significance (usually 0.05).
* Power: the probability of correctly rejecting a false null hypothesis.

The “*pwr”* [24], “*samplesizeestimator”* [25], and “*presize”* [26] packages may be used for power - sample size estimation in the examples below.

### Power, Sample Size for Comparing Continuous Variables

The following example uses t-test for comparing continuous variable between two groups: A researcher is testing a new drug to lower cholesterol. They hypothesize the drug will lower LDL cholesterol by 15 mg/dL compared to a placebo. The standard deviation of LDL cholesterol is known to be 30 mg/dL.

The sample size needed for 80% power (using *pwr.t.test()* of the “*pwr*” package [24]):

*#library(pwr)*

*# Calculate effect size (Cohen's d)*

effect\_size\_d <- 15 / 30 *# Mean difference / standard deviation*

*# Calculate sample size*

pwr.t.test(d = effect\_size\_d, sig.level = 0.05, power = 0.80,

type = "two.sample")

##

## Two-sample t test power calculation

##

|  |  |  |  |
| --- | --- | --- | --- |
| ## | n | = | 63.76561 |
| ## | d | = | 0.5 |
| ## | sig.level | = | 0.05 |
| ## | power | = | 0.8 |
| ## | alternative | = | two.sided |
| ## |  |  |  |

## NOTE: n is number in \*each\* group

The output shows that approximately 64 participants are needed in each group and a total of 128 participants is needed for the study.

For some reasons, the study stops after recruiting 40 subjects per group, what is the power for detecting the difference?

pwr.t.test(n = 40, d = 0.5, sig.level = 0.05, type = "two.sample")

|  |  |  |
| --- | --- | --- |
| ## |  | |
| ## | Two-sample | t test power calculation |
| ## |  |  |
| ## | n | = 40 |
| ## | d | = 0.5 |
| ## | sig.level | = 0.05 |
| ## | power | = 0.5981469 |
| ## | alternative | = two.sided |
| ## |  |  |

## NOTE: n is number in \*each\* group

The output shows that with 40 subjects per group, the power drops to 0.6.

### Power, Sample Size for Comparing Categorical Variables

The example below uses chi-square test for comparing categorical variable between two groups: A city wants to test a new educational campaign to increase flu vaccination rates. The current rate (p\_control) is 40%. They will consider the campaign a success if it raises the rate to 55% (p\_campaign). We can calculate the sample size needed using *pwr.2p.test()* after estimating effect size using *ES.h()*:

*# Calculate effect size h*

effect\_size\_h <- ES.h(p1 = 0.55, p2 = 0.40)

effect\_size\_h

## [1] 0.3015253

*# Calculate sample size per group*

pwr.2p.test(h = effect\_size\_h,

sig.level = 0.05,

power = 0.80,

alternative = "two.sided")

##

## Difference of proportion power calculation for binomial distribution (arcsine transformation)

##

## h = 0.3015253

## n = 172.6589

## sig.level = 0.05

## power = 0.8

## alternative = two.sided ##

## NOTE: same sample sizes

The output show that approximately 170 participants are needed in each group (total N = 340) to have an 80% chance of detecting a true increase in vaccination rate from 40% to 55%.

### Sample Size for Estimating Parameter with Desired Precision

This is a different type of sample size calculation. Here, the goal is not to test a hypothesis but to estimate a population parameter (like a mean or proportion) with a pre-specified level of precision E (e.g., a narrow confidence interval).

#### Estimating Mean

The example below estimates the mean cholesterol level in a specific patient population. Based on prior literature, they expect the standard deviation (*σ*) to be 40 mg/dL. They want their final estimate to be accurate within ± 5 mg/dL with 95% confidence.

The package “*samplesizeestimator”* may be used for estimating mean with absolute precision [25].

*#library(samplesizeestimator)*

sample\_size <- estm(sig = 40, prec = 5, alp = 0.05,

relative = FALSE) *#alpha = 0.05 corresponding to 95%CI*

print(sample\_size)

## $Sample\_Size

## [1] 246

##

## [[2]]

## [1] " Description: Sample size is estimated with an expected "

## [2] "standard deviation of the 'outcome of interest' among the "

## [3] "'population' as 40 units at 5% level of significance with "

## [4] " an absolute precision of 5 units, the estimated sample "

## [5] " size is 246. "

The output shows that 246 patients are needed to be 95% confident that the true population mean cholesterol is within 5 mg/dL of their sample mean.

#### Estimating Proportion

The example below estimates the proportion of patients who are “very satisfied” with care in a hospital. The estimate should be within ± 4% with 95% confidence. There is no prior information about the true proportion.

The package “*presize”* [32] may be used with various method options (using *prec\_prop()*). Note that, if no best guess for the proportion exists, use p = 0.5, as this is the most conservative choice and yields the largest required sample size.

*#library(presize)*

prec\_prop(p = 0.5, conf.width = 0.08, conf.level = 0.95,

method = "wald") *# CI width = margin of error (0.04) \*2*

##

## sample size for a proportion with Wald confidence interval.

##

## p padj n conf.width conf.level lwr upr

## 1 0.5 0.5 600.2279 0.08 0.95 0.46 0.54

##

## NOTE: padj is the adjusted proportion, from which the ci is calculated.

The output shows that 601 patients are needed to be 95% confident that the true proportion of “very satisfied” patients is within 4 percentage points of their survey result.

One may calculate the expected confidence interval width for a given sample

size using *prec\_prop()*.

prec\_prop(p = 0.5, n = 601, method = "wald") ##

## precision for a proportion with Wald confidence interval.

##

## p padj n conf.width conf.level lwr upr

## 1 0.5 0.5 601 0.0799486 0.95 0.4600257 0.5399743

##

## NOTE: padj is the adjusted proportion, from which the ci is calculated.

## Chapter Summary

* Examining the data distribution is important before performing statistical tests
* T-test (*t.test()*) (normal assumption required) and its non-parametric alternatives (*wilcox.test()*, *kruskal.test()*) are the common statistical tests used for testing continuous variables
* Chi-square test (*chisq.test())* and Fisher’s exact tests (*fisher.test())* are the

common statistical tests used for testing categorical variables

* Power - sample size analysis may be done with the “*pwr”*, “*samplesizeestimator”*, and “*presize”* packages

# Linear Regression with R

This chapter provides a basic guide to fitting and interpreting Simple and Multiple Linear Regression models for bio-medical data using R. The chapter begins by introducing basic concepts of the linear model, including its core formula and key assumptions such as linearity, independence, and homoscedasticity. The practical focus of the chapter centers on the *lm()* function in R for fitting linear models. Readers will also learn how to interpret the model output, perform essential model diagnostic checks, model selection and model validation.

## Introduction to Linear Regression

Linear regression evaluates the relationship between a continuous dependent variable (outcome) and one or more independent variables (predictors). From a sample data, linear regression is used to estimate population parameters characterizing these relationships and their uncertainties.

Linear regression has two primary applications:

* Estimating Effects: It helps estimate the specific effect of a predictor on the outcome, particularly while controlling for other variables. This is a key strength of multiple linear regression, which allows you to statistically estimate the impact of one variable (e.g., smoking) while controlling the confounding influence of others (e.g., age or weight).
* Prediction: It helps predict outcomes for future observations. Once a model is built and validated, you can input the predictor values for a new individual and the model will forecast the most likely value for their outcome variable.

First, we load the main R packages used in this chapter *("knitr" [5], "rmarkdown" [6], "tidyverse" [11], "arsenal" [19], "kableExtra" [12], "yardstick" [33]*):

*#load multiple packages*

Packages <- c("knitr", "rmarkdown", "tidyverse", "arsenal",

"kableExtra", "yardstick")

lapply(Packages, library, character.only = TRUE)

## Simple Linear Regression

### Simple Linear Model with R

Simple linear regression quantifies the relationship between one outcome and one predictor. Below is a generated example data:

|  |  |  |  |
| --- | --- | --- | --- |
| ## |  | x | y |
| ## | 1 | 3.620757 | 0.8449674 |
| ## | 2 | 3.035641 | 0.8304951 |
| ## | 3 | 3.773154 | 0.9622945 |
| ## | 4 | 4.272489 | 0.9096735 |
| ## | 5 | 3.370975 | 0.8072324 |
| ## | 6 | 2.837146 | 0.8259437 |

In which:

* + - * y is outcome or response or dependent variable (DV)
      * x is predictor or regressor or independent variable (IV)

Simple linear regression assumes the linear relationship between x and y. The formula for simple linear regression model may be written as:

*y* = *β*0 + *β*1*x* + *ϵ*

In which:

* + - * *β*0 is the intercept or the predicted *y* when *x* = 0
      * *β*1 is the slope or the predicted change in *y* for a one-unit increase in *x*
      * *ϵ* is the error term or the deviation of the predicted *y* value from the observed

*y* [34].

Simple linear model may be fitted with base R using *lm()* as below:

slm.fit <- lm(y ~ x, data = df)

The *lm()* returns complex objects which may be examined by:

* + - * *coef()*: model coefficients
      * *summary()*: coefficients, standard errors, p-values, model F-test, etc.
      * *predict()*: model predictions
      * *residuals()*: various kinds of residuals
      * *confint()*: confidence intervals for coefficients
      * *plot()*: diagnostic plots

The result output may be summarized as below:

summary(slm.fit)

|  |  |
| --- | --- |
| ##  ## ## | Call:  lm(formula = y ~ x, data = df) |
| ## |  |
| ## | Residuals: |
| ## | Min 1Q Median 3Q Max |
| ## | -0.25771 -0.07126 0.00101 0.05635 0.32485 |
| ## |  |
| ## | Coefficients: |
| ## | Estimate Std. Error t value Pr(>|t|) |
| ## | (Intercept) 0.525215 0.019149 27.43 <0.0000000000000002 \*\*\* |
| ## | x 0.111954 0.006056 18.48 <0.0000000000000002 \*\*\* |
| ## | --- |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 |
| ## | Residual standard error: 0.1034 on 298 degrees of freedom |
| ## | Multiple R-squared: 0.5342, Adjusted R-squared: 0.5326 |
| ## | F-statistic: 341.7 on 1 and 298 DF, p-value: < 0.00000000000000022 |

The *summary()* output shows :

* + - * Model formula
      * Summary of residual distribution
      * Estimated coefficients, standard errors, t-values, and p-values (for testing a hypothesis that a coefficient equals to zero).
      * A quick visual shortcut to p-values by \*
      * Residual standard error measuring the average amount that observed responses deviate from the fitted regression line.
      * Degrees of freedom (*d f* ) represent the number of independent observations

(*N*) in the data that are available to estimate a parameter. For linear regression,

*p* = the number of predictors +1 and *d f* = *N* − *p*.

* + - * R-square (*R*2), a measure to test if the model is a good fit, shows the amount of variance of y explained by x.

*R*2 = *Explained variation o f the model* / *Total variation o f the model*

If the model fit the data well, *R*2 is near 1 and if the model fit the data poorly, *R*2 is near 0. *Adjusted R*2 shows the same as *R*2 but adjusted by the number of cases and number of variables. When the number of variables is small and the number of outcomes is very large then *Adjusted R*2 is closer to *R*2.

* + - * *F*-statistic: indicates whether the overall model is statistically significant (at

least one predictor is related to the outcome)

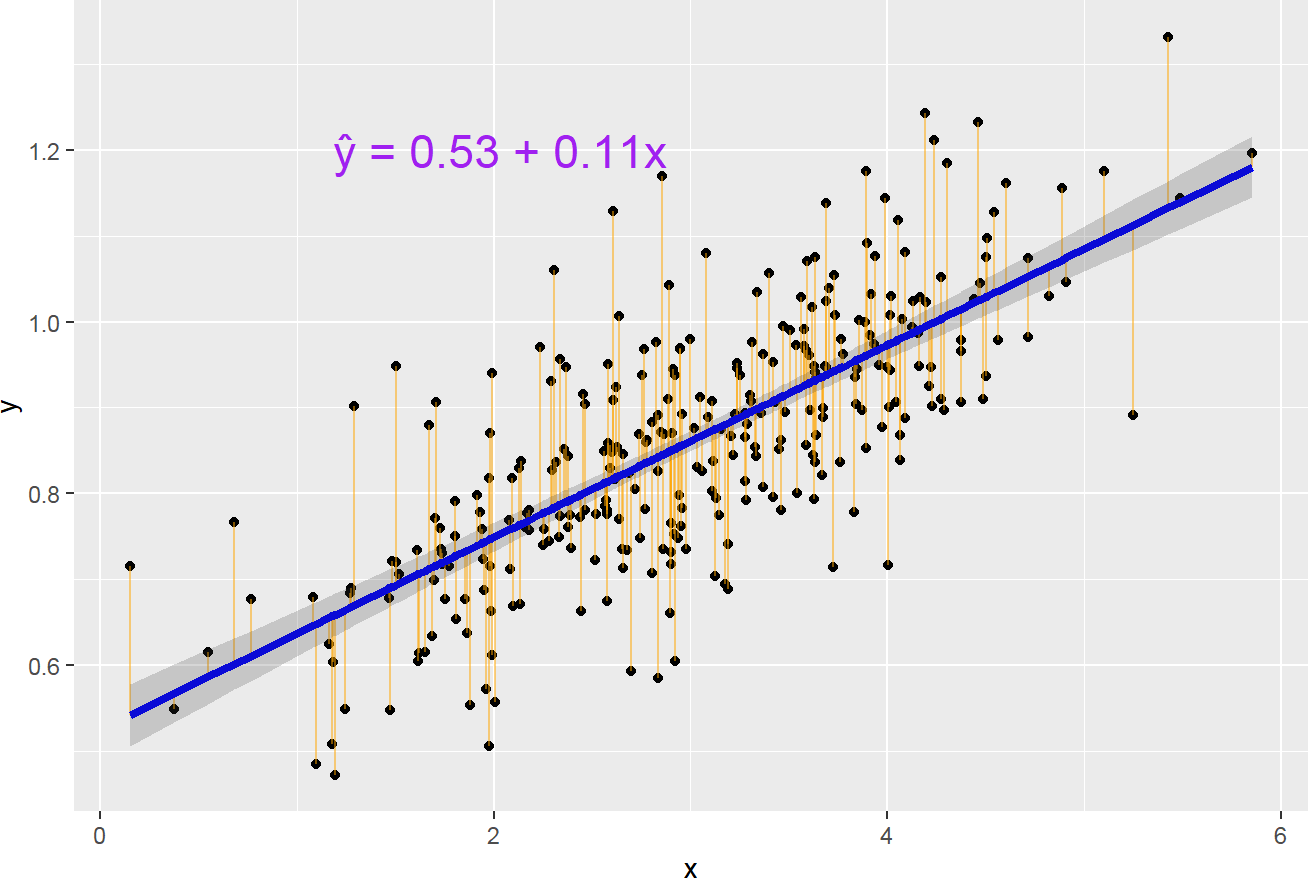
The *summary()* output above show that x is positively and significantly associated with y. One unit increase in x is associated with 0.1119536 increase of y.

Confidence intervals may be computed with *confint()*:

|  |  |  |
| --- | --- | --- |
| ci ci | <- confint(slm.fit) |  |
| ## | 2.5 % | 97.5 % |
| ## | (Intercept) 0.4875300 | 0.5628993 |
| ## | x 0.1000348 | 0.1238724 |

The output shows that we are 95% confident that the true population slope is between 0.100 and 0.124. Critically, this interval does not contain 0. This reinforces the p-value's conclusion that *x* has a statistically significant, positive effect on *y*.

The intercept, the slope and residuals are visualized as below:



Source: Figure by author(s).

**Figure 31**. Intercept, Slope and Residuals Plot.

The straight blue line in the image above represents the predicted values. The model fit may be assessed by plotting the predicted values to evaluate if they look reasonable against the observed data. The shaded area represents the confidence interval of predicted values.

The orange vertical line from the straight line to the observed data value is the residual. For a good model, the sum of the residuals should be approximately zero or as low as possible and the residual plot should look random. If there is pattern in the residual plot, the simple linear model may not fit the data well and there may be a need to transform the variable (e.g., adding quadratic, or cubic term,. . . ).

The residuals may be plotted in R using *plot()* as below:

plot(slm.fit$residuals)



Source: Figure by author(s).

**Figure 32**. Residuals plot.

The residual plot should be a random, horizontal cloud of points centered around 0. The plot above shows exactly this, with no obvious trend or pattern. This suggests our simple linear model is a reasonable fit for the data.

### Linear Model Assumptions

Below are assumptions for inferences made from a regression model [34].

* + - * **L**: **Linear relationship** between the mean outcome (Y) and the predictor variable (X)
      * **I**: The errors are **independent**
      * **N**: The outcomes are **normally distributed** at each level of predictor variable (X)
      * **E**: The variance of the outcomes is **equal** for all levels of predictor variable (X) (Homoscedasticity)

If the assumptions are violated, the model estimates and inferences may not be valid.

### Checking Linear Model Assumptions

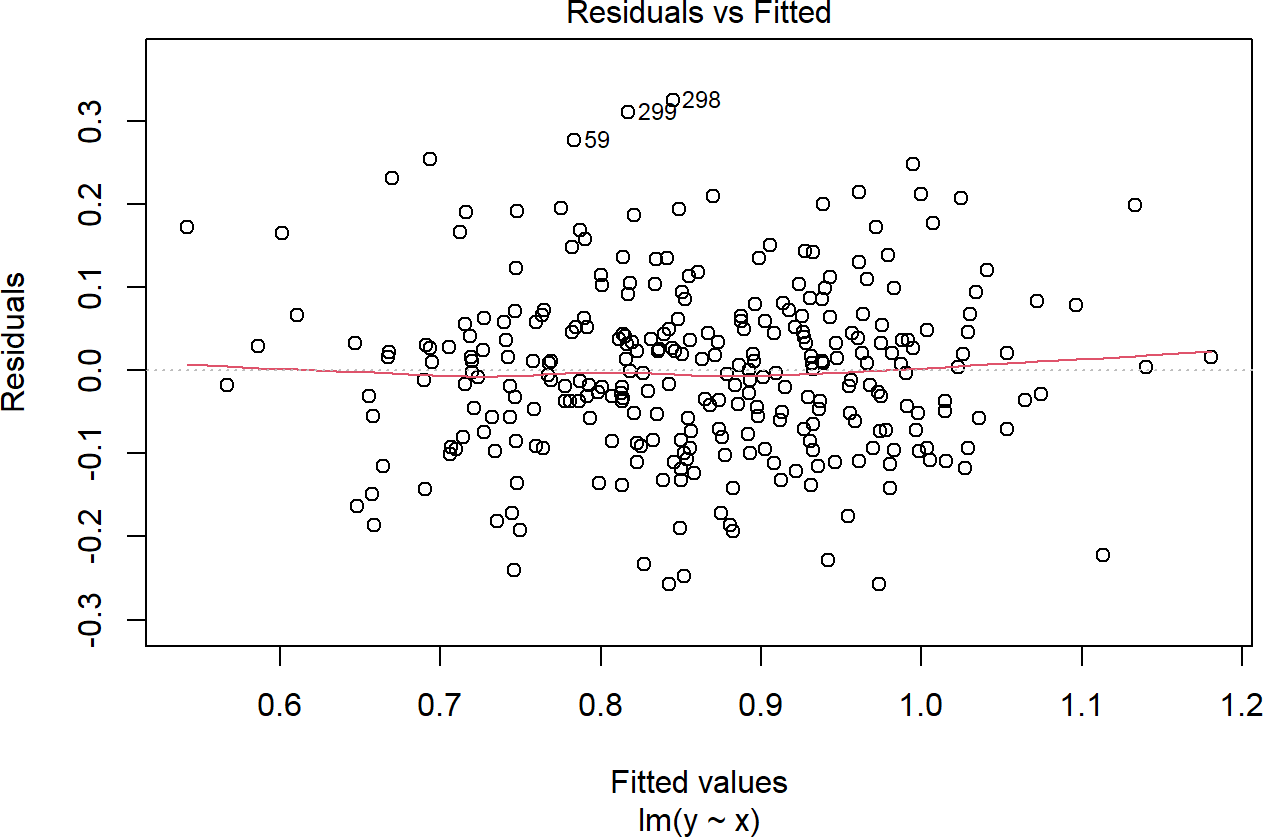
#### Linear Assumption

If linear assumption is met, the residuals should be random and there should be no systematic trend or pattern in residual plot.

Diagnostic plots may be done by *plot()* on the *lm* object. The first plot is the residuals vs fitted values plot.

*# which = 1 plots residuals vs.fitted values plot*

plot(slm.fit, which = 1)

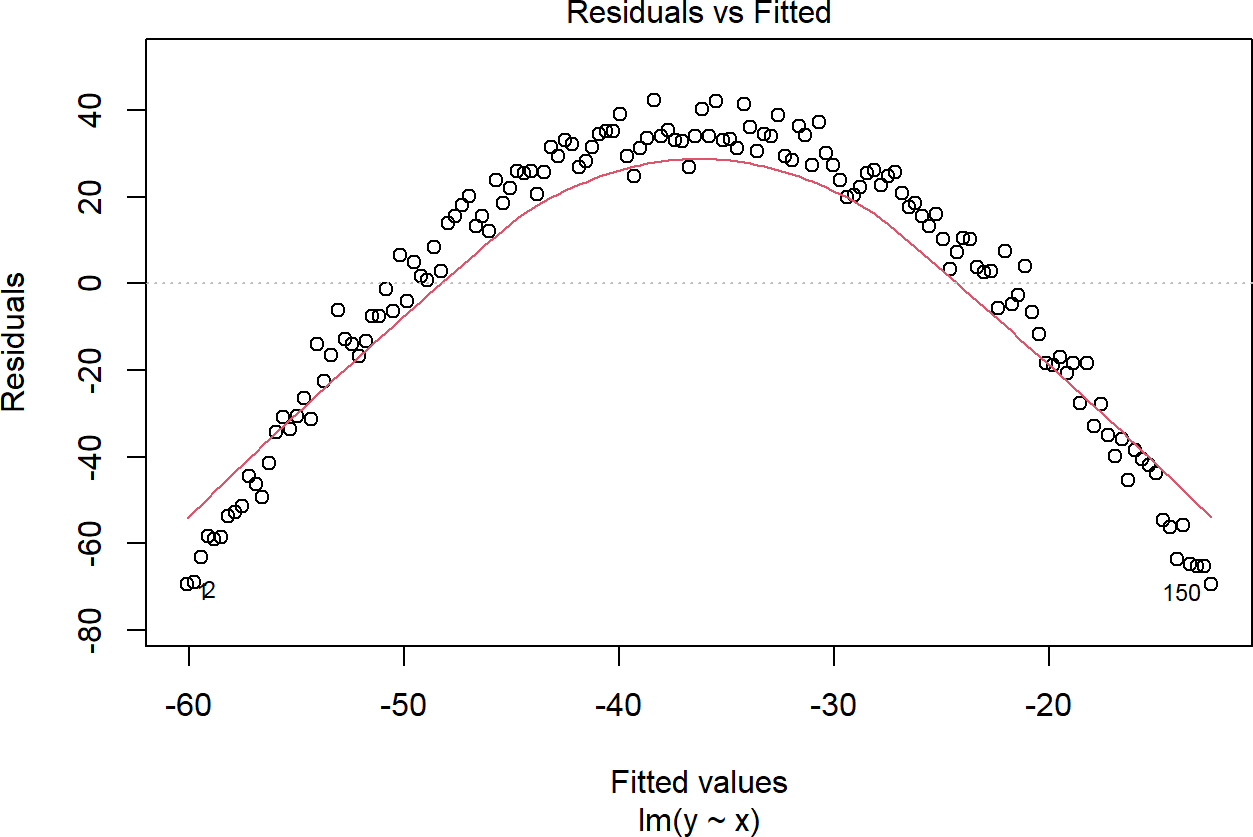


Source: Figure by author(s).

**Figure 33**. Residuals vs fitted values plot.

In our example, there is no trend in residual plot. If there is a trend in the residuals, it suggests a nonlinear relationship between predictor and outcome variables. In this case, one may consider adding non-linear term to the model (e.g., cubic, quadratic, splines,. . . ) or using non-linear transformations (e.g., logarithm) to either predictor or outcome variables.

Below is an example plot showing a clear trend of residuals:



Source: Figure by author(s).

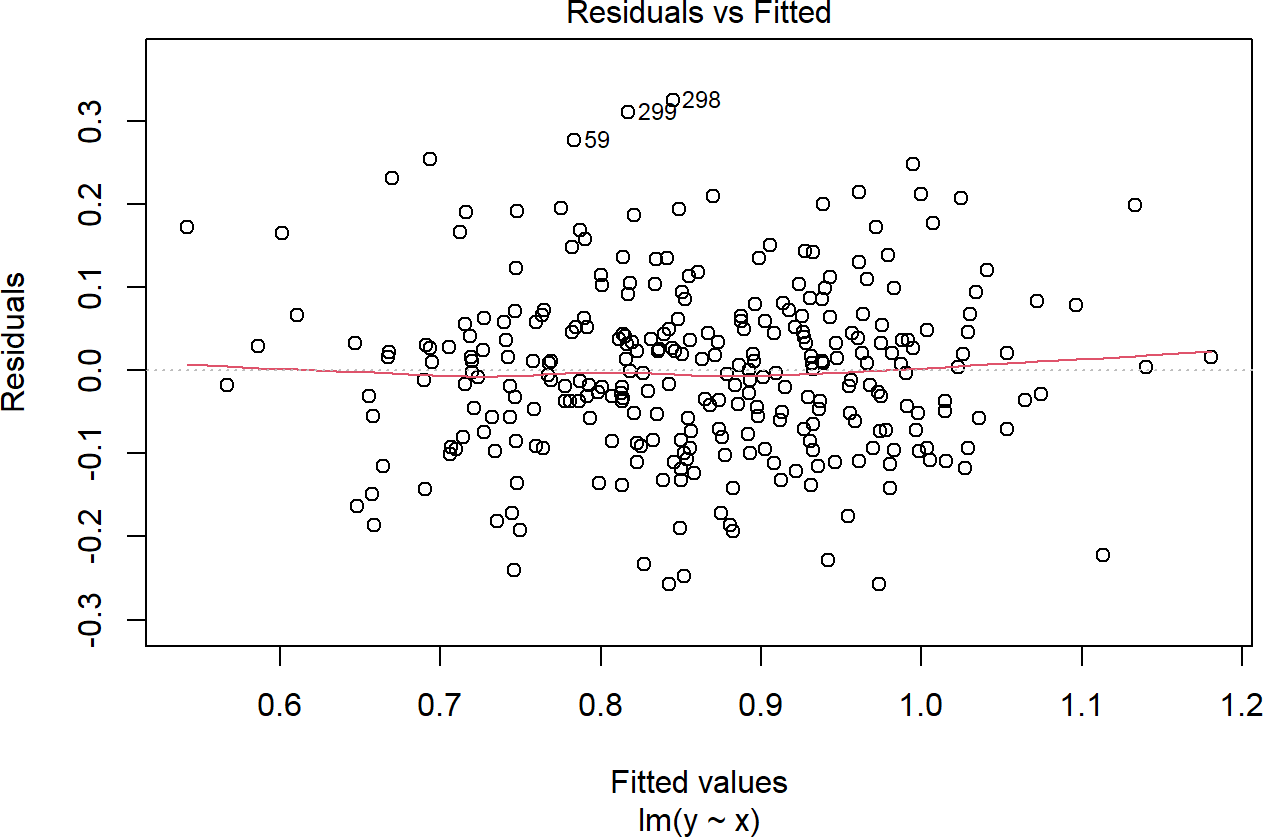
**Figure 34**. Example trend of residuals.

#### Equal Variance Assumption (Homoscedasticity)

If this assumption is violated, standard errors will be inaccurate resulting in inaccurate test statistics, confidence intervals and p-values. For this assumption, the diagnostic plot of residuals vs. fitted values should show random scatter around the zero line with equal spread across the range of fitted values as in the residual plot of our example.

*# which = 1 plots residuals vs.fitted values plot*

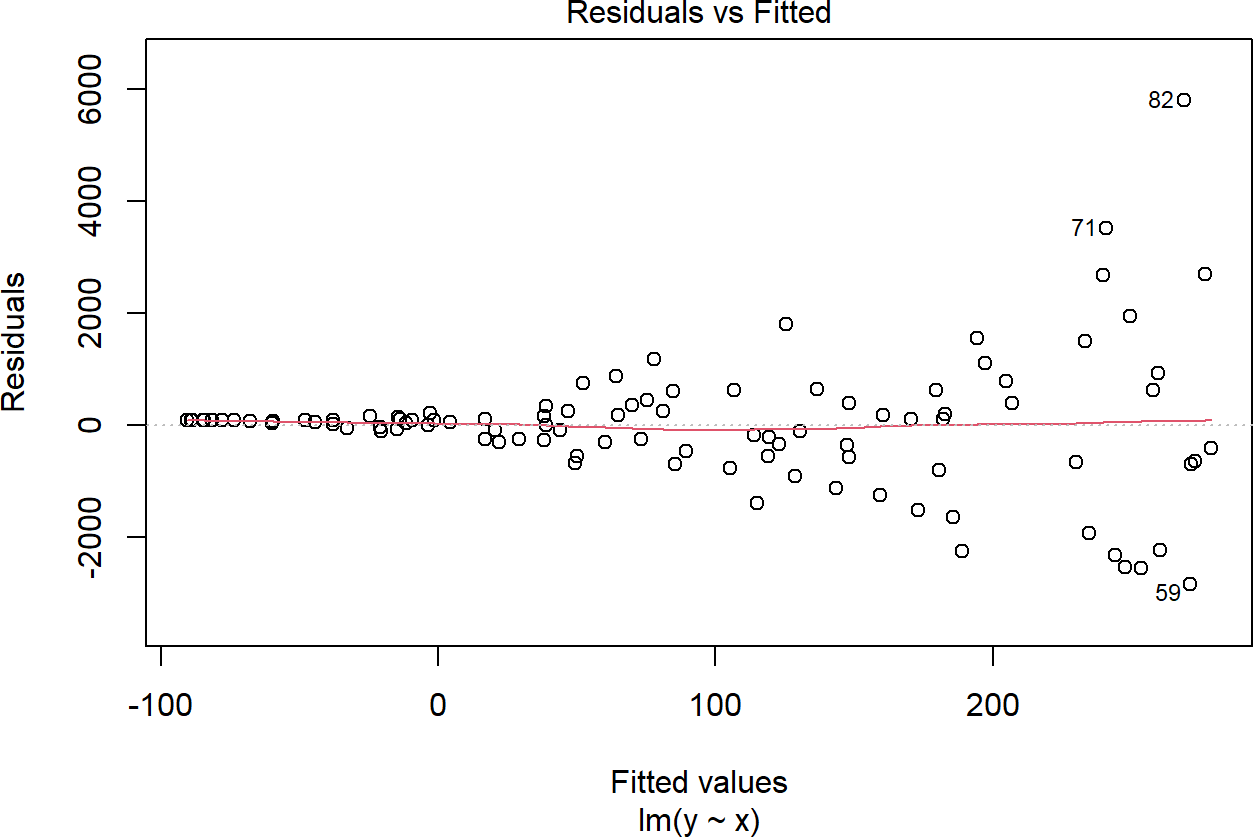
plot(slm.fit, which = 1)



Source: Figure by author(s).

**Figure 35**. Residual plot: equal variance.

Unequal variance or heteroscedasticity is often observed in the form of increasing variance of the residuals with increasing fitted values as shown in the figure below:



Source: Figure by author(s).

**Figure 36**. Residual plot: unequal variance.

If equal variance assumption is violated, one may consider:

* Transformation of variables (e.g., log transformation of the outcome variable)
* Using robust standard errors (heteroscedastic-consistent or “sandwich”) (may be done with R package “*sandwich*” [35])
* Using a mixed model for heteroscedasticity modeling (may be done with R package “*nlme*” [36])

#### Independence Assumption

Independence assumption is not typically checked with a plot but is assessed based on the study design. If this assumption is violated, it may affect the accuracy of confidence intervals and p-values.

Common causes of data dependency are:

* Sampling in clusters
* Repeated measurements of the same subjects

The approaches to handle this assumption violation are:

* Model data dependency with mixed/multilevel models, fixed effect models,

. . . (may be done with R package “*lme4*” [37])

* Using cluster robust standard errors

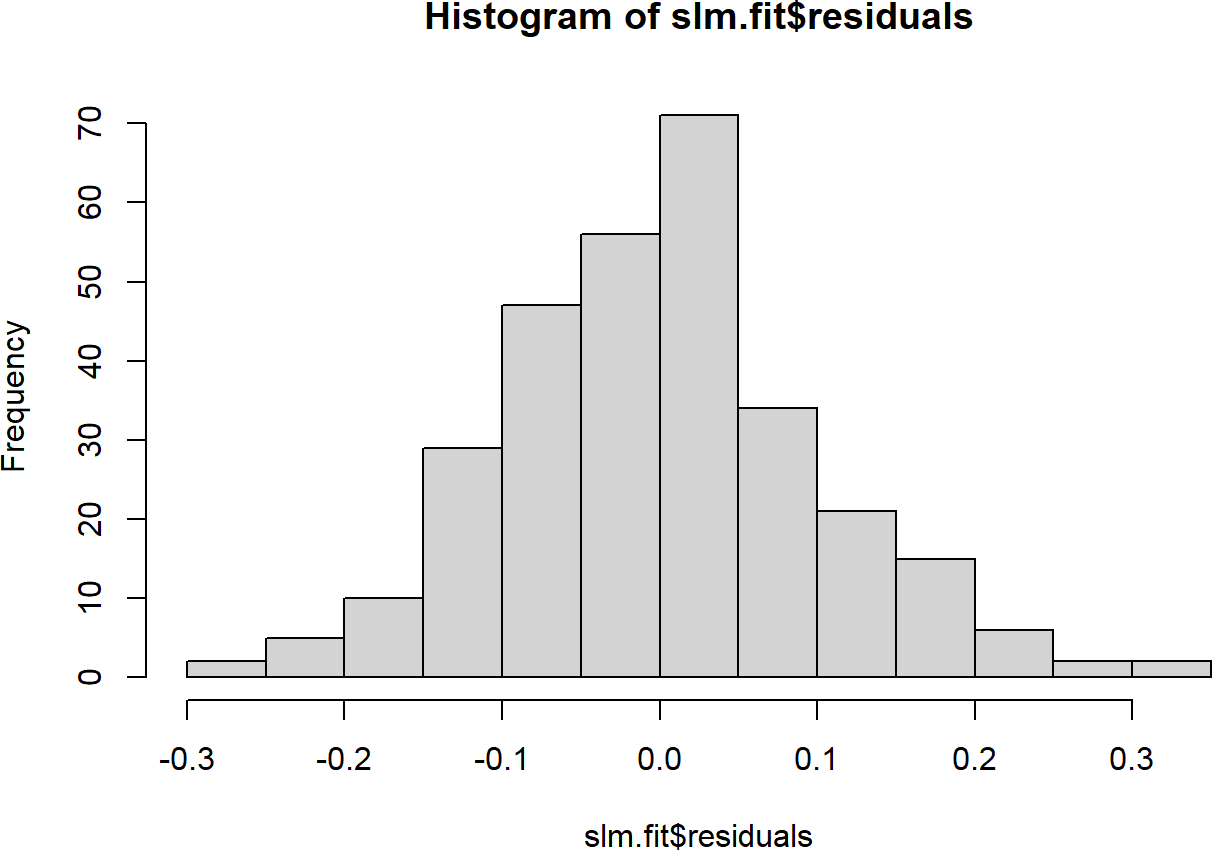
#### Normal Distribution Assumption

If the errors are not normally distributed, the accuracy of confidence interval and p-values may be affected, especially in case of small sample size (the impact is less with large sample size).

Histogram and Q-Q plot of residuals may be used as diagnostic plots:

*# A simple histogram of residuals*

hist(slm.fit$residuals, breaks = 20)



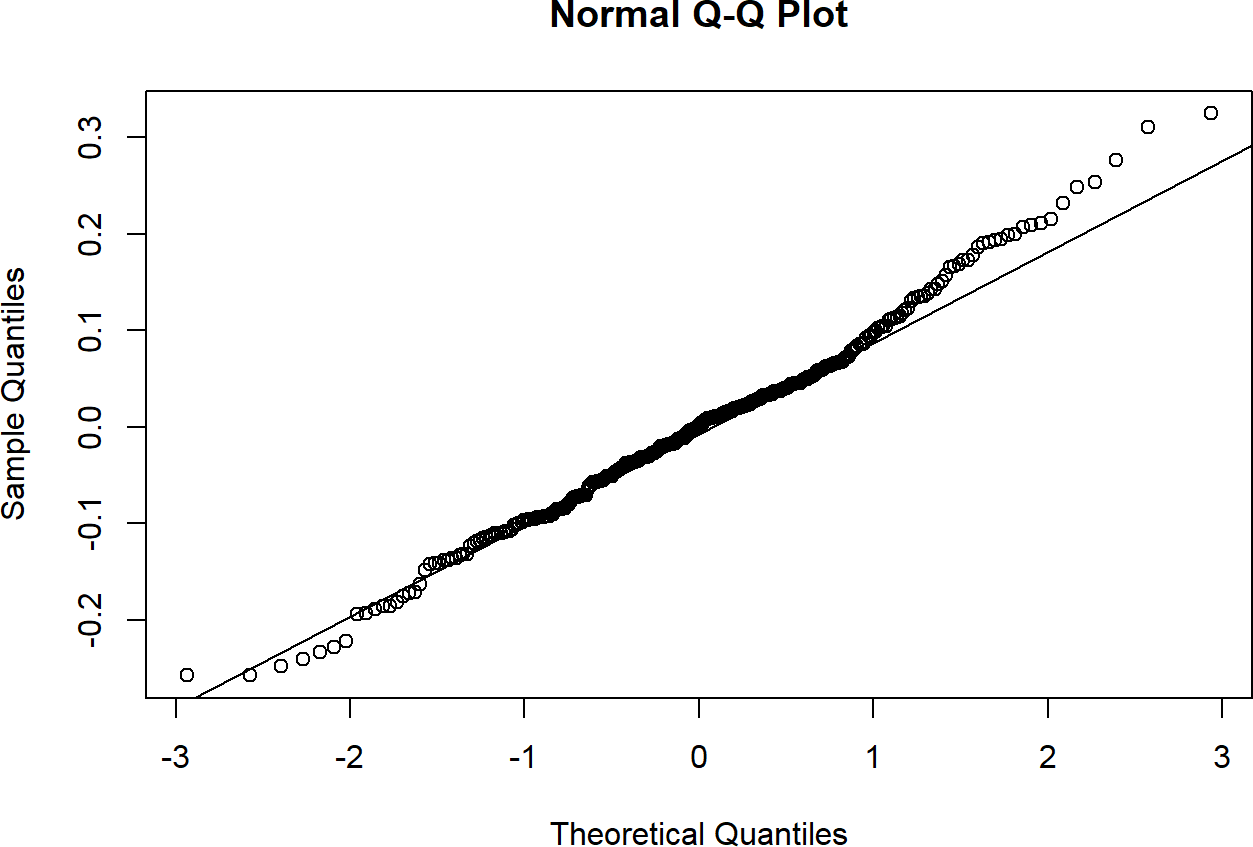
Source: Figure by author(s).

**Figure 37**. Histogram normal distribution.

*# Normal Quantile to Quantile plot*

qqnorm(slm.fit$residuals)

qqline(slm.fit$residuals)



Source: Figure by author(s).

**Figure 38**. Q-Q plot normal distribution.

To support the normality assumption, the histogram should show a symmetric, bell-shaped curve and Q-Q plot should display the points falling closely along the straight diagonal line as seen in the figures above.

If the errors are not normally distributed, the Q-Q plot will show deviation from the diagonal line. The approaches to handle this issue may be:

* Transformation (e.g., logarithm,. . . ) of the outcome variable
* Using generalized linear models (may be done with *glm()* function)

### Checking Influential Observations

Influential observations change model estimates substantially when included vs when excluded from the data. Model coefficients change when a particular observation is excluded may be evaluated by influence diagnostic measures:

* + - * *dfbeta*measures how a specific coefficient changes when deleting an observation
      * Cook’s distancemeasures how much all coefficients change when deleting an observation

Influence diagnostic measures may be done with *influence.measures()* on an *lm* object, which results in several influence measures including *dfbetas*for each coefficient and Cook’s distance.

If there are any unusually large influence measures, it will be flagged by an \* in the *infl* column.

influence <- influence.measures(slm.fit)

str(influence)

## List of 3

## $ infmat: num [1:300, 1:6] 0.01358 -0.00544 -0.00353 0.04819

0.00224 ...

## ..- attr(\*, "dimnames") = List of 2

## .. ..$ : chr [1:300] "1" "2" "3" "4" ...

## .. ..$ : chr [1:6] "dfb.1\_" "dfb.x" "dffit" "cov.r" ...

## $ is.inf: logi [1:300, 1:6] FALSE FALSE FALSE FALSE FALSE FALSE ...

## ..- attr(\*, "dimnames") = List of 2

## .. ..$ : chr [1:300] "1" "2" "3" "4" ...

## .. ..$ : chr [1:6] "dfb.1\_" "dfb.x" "dffit" "cov.r" ...

## $ call : language lm(formula = y ~ x, data = df)

## - attr(\*, "class") = chr "infl"

### Categorical Predictor Variables

Categorical predictor variables with k (>= 2) categories will have k-1 coefficient estimates. The simplest categorical variable is dummy variable with 2 categories 0 and 1 denoting not belonging or belonging to a category. An example of this dummy variable may be “smoker” with category “no” (equivalent to 0) and “yes” (equivalent to 1). Categorical variables with k categories is treated as k-1 dummy variables with the omitted category treated as reference category. Categorical variables in R are represented by factor variables and the factor order and reference group may be selected. When performing *lm()*, R will automatically create dummy variables for each level of factor except for the first level (which is used as the reference group).

#### Example Simple Linear Regression with Categorical Predictor Variable

We will use [example data regarding the association of medical costs and patient](https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/insurance.csv) [information from github.](https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/insurance.csv) We first load the data and summarize all variables with the “*arsenal*” package [19].

*# Load example linear regression data*

url\_base<-"https://raw.githubusercontent.com/stedy/"

data\_dir<-"Machine-Learning-with-R-datasets/master/insurance.csv"

dlm<-read.csv(paste0(url\_base,data\_dir))

colnames(dlm)<-tolower(colnames(dlm))

str(dlm)

## ‘data.frame’: 1338 obs. of 7 variables: $ age : int 19 18 28 33 32 31 46 37 37 60 . . .

## $ sex : chr “female” “male” “male” “male” . . . $ bmi : num 27.9 33.8 33 22.7 28.9 .

## . . $ children: int 0 1 3 0 0 0 1 3 2 0 . . . $ smoker : chr “yes” “no” “no” “no” . . . $

## region

## : chr “southwest” “southeast” “southeast” “northwest” . . . $ charges : num

## 16885 1726 4449 21984 3867 . . .

*#turn character variables into factor variables*

dlm[,colnames(dplyr::select\_if(dlm, is.character))] <- lapply(dlm[,colnames(dplyr::select\_if(dlm, is.character))],

as.factor)

*#summary of all variables in the dataset*

my\_controls <- tableby.control(

numeric.stats = c("meansd", "medianq1q3","meanCI","range","Nmiss2"), cat.stats = c("countpct", "Nmiss2"),

stats.labels = list( meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)", range = "Min - Max",

Nmiss2 = "Missing", meanCI = "Mean (95%CI)"

)

)

mylabels <-as.list(colnames(dlm)) names(mylabels)<-colnames(dlm)

tabs <- tableby(as.formula(paste("",paste(colnames(dlm),

collapse = "+"),sep = "~")),

data = dlm,

control = my\_controls) kable(summary(tabs,labelTranslations = mylabels, text = TRUE),

caption = "Data summary table", longtable = TRUE)%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position", "repeat\_header"))

**Table 7**. Data summary table.

|  |  |
| --- | --- |
|  | **Overall (N = 1338)** |
| age |  |
| - Mean (SD) | 39.207 (14.050) |
| - Median (Q1, Q3) | 39.000 (27.000, 51.000) |
| - Mean (95%CI) | 39.207 (38.454, 39.961) |
| - Min−Max | 18.000−64.000 |
| - Missing | 0 |
| sex |  |
| - female | 662 (49.5%) |
| - male | 676 (50.5%) |
| - Missing | 0 |
| bmi |  |
| - Mean (SD) | 30.663 (6.098) |

|  |  |
| --- | --- |
|  | **Table 7.** *Cont.* |
|  | **Overall (N = 1338)** |
| - Median (Q1, Q3) | 30.400 (26.296, 34.694) |
| - Mean (95%CI) | 30.663 (30.336, 30.990) |
| - Min - Max | 15.960−53.130 |
| - Missing | 0 |
| children |  |
| - Mean (SD) | 1.095 (1.205) |
| - Median (Q1, Q3) | 1.000 (0.000, 2.000) |
| - Mean (95%CI) | 1.095 (1.030, 1.160) |
| - Min−Max | 0.000−5.000 |
| - Missing | 0 |
| smoker |  |
| - no | 1064 (79.5%) |
| - yes | 274 (20.5%) |
| - Missing | 0 |
| region |  |
| - northeast | 324 (24.2%) |
| - northwest | 325 (24.3%) |
| - southeast | 364 (27.2%) |
| - southwest | 325 (24.3%) |
| - Missing | 0 |
| charges |  |
| - Mean (SD) | 13,270.422 (12,110.011) |
| - Median (Q1, Q3) | 9382.033 (4740.287, 16,639.913) |
| - Mean (95%CI) | 13,270.422 (12,620.954, 13,919.890) |
| - Min−Max | 1121.874−63,770.428 |
| - Missing | 0 |

Source: Table by author(s).

We then examine data distribution with histogram:

*# Get numeric variables*

numeric\_vars <- colnames(dplyr::select\_if(dlm, is.numeric))

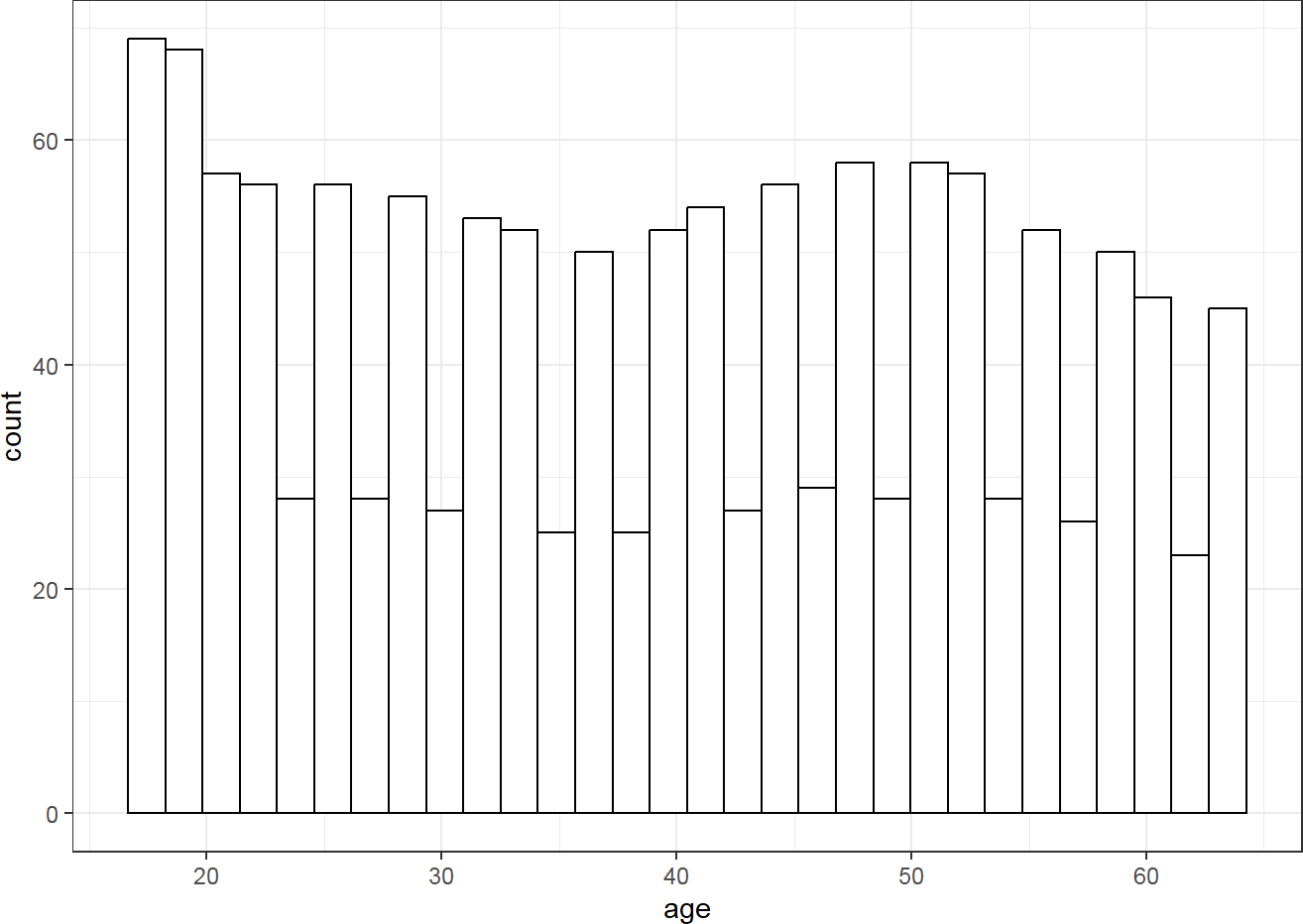
*# Create a loop the do histogram for each of all numeric variables*

for (i in 1:length(numeric\_vars)) {

print(ggplot(data = dlm, aes(x = dlm[, numeric\_vars[i]])) + geom\_histogram(colour = "black", fill = "white") + theme\_bw() +

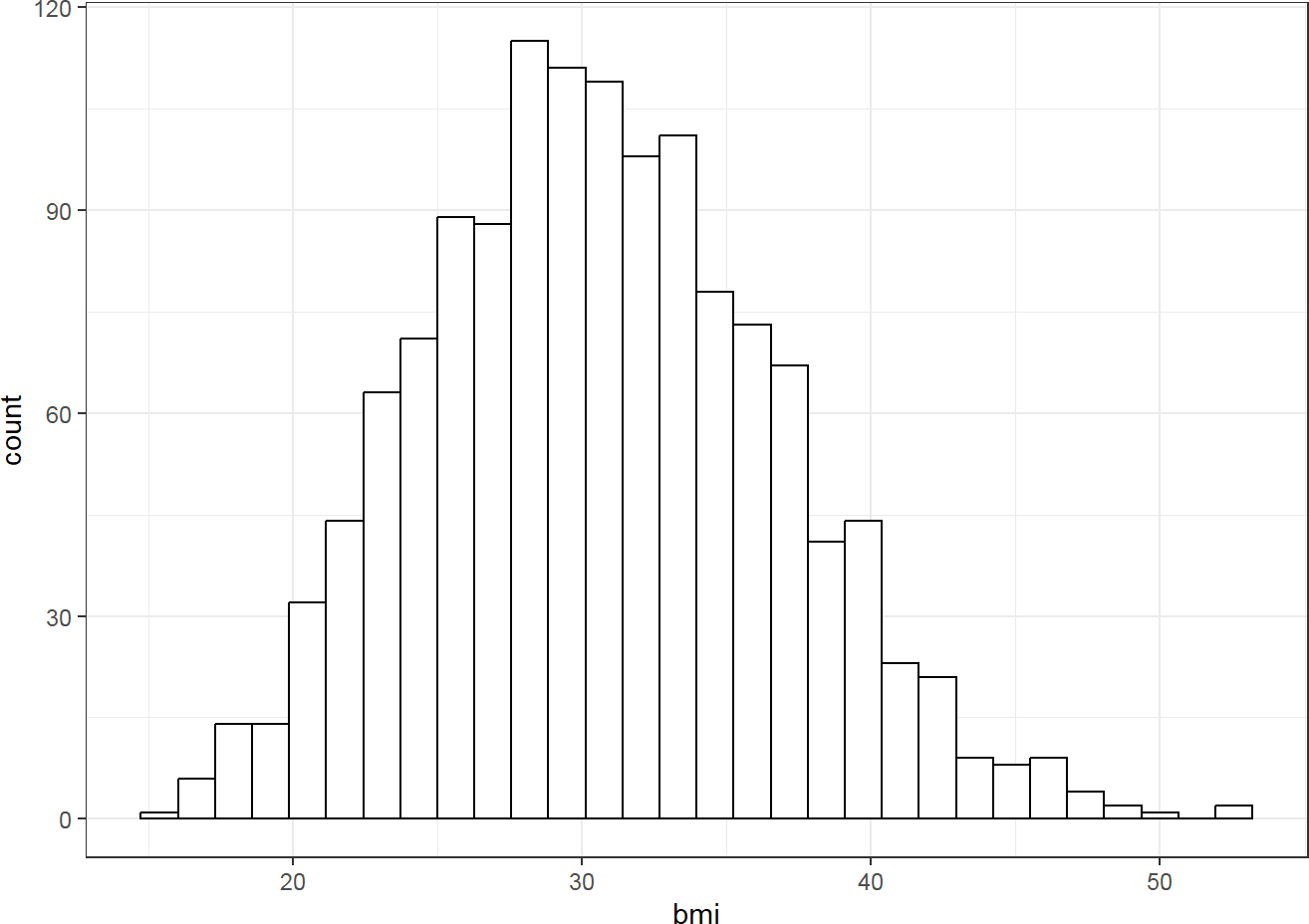
labs(x = numeric\_vars[i]))

}



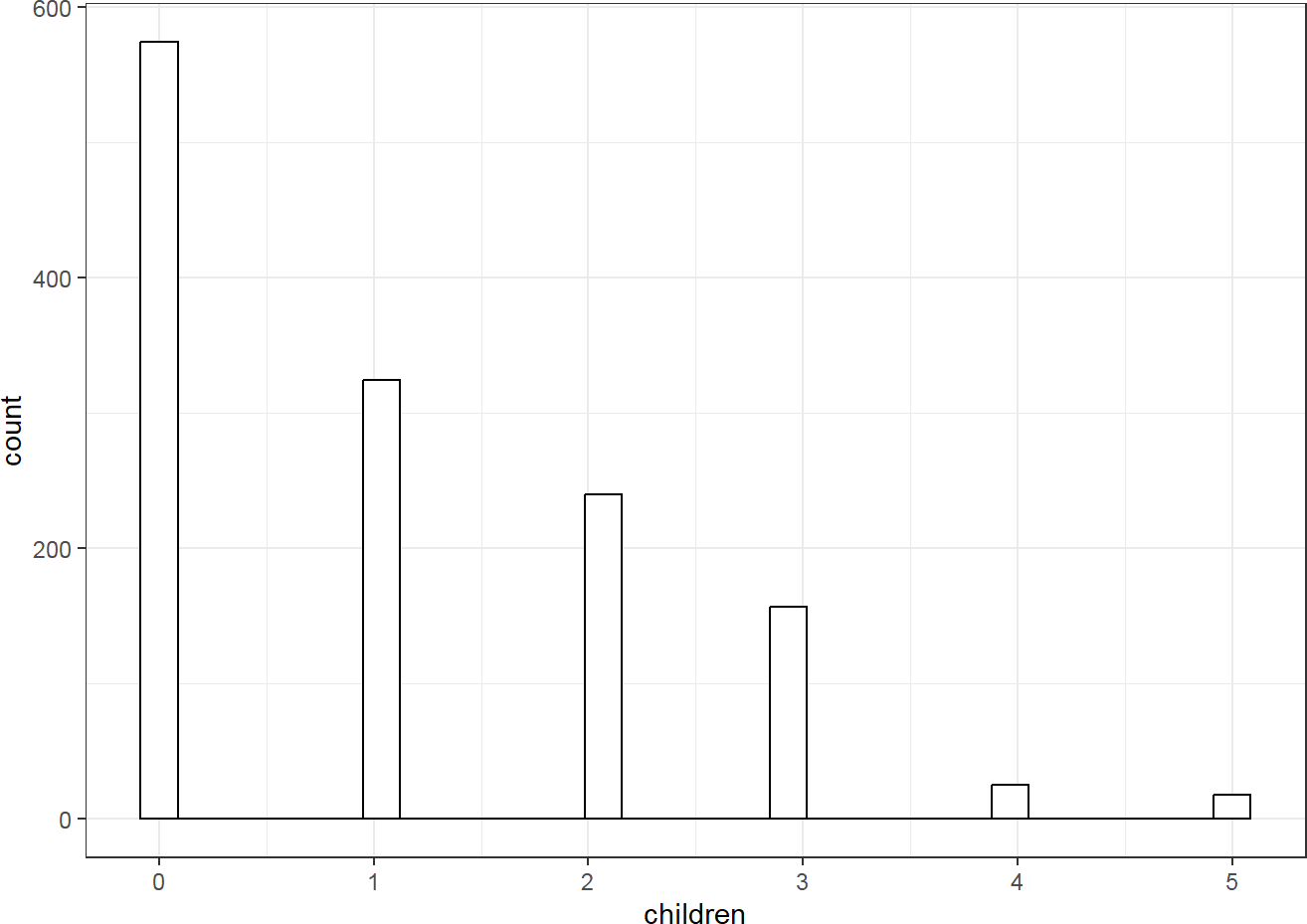
Source: Figure by author(s).

**Figure 39**. Histogram of continuous variable.



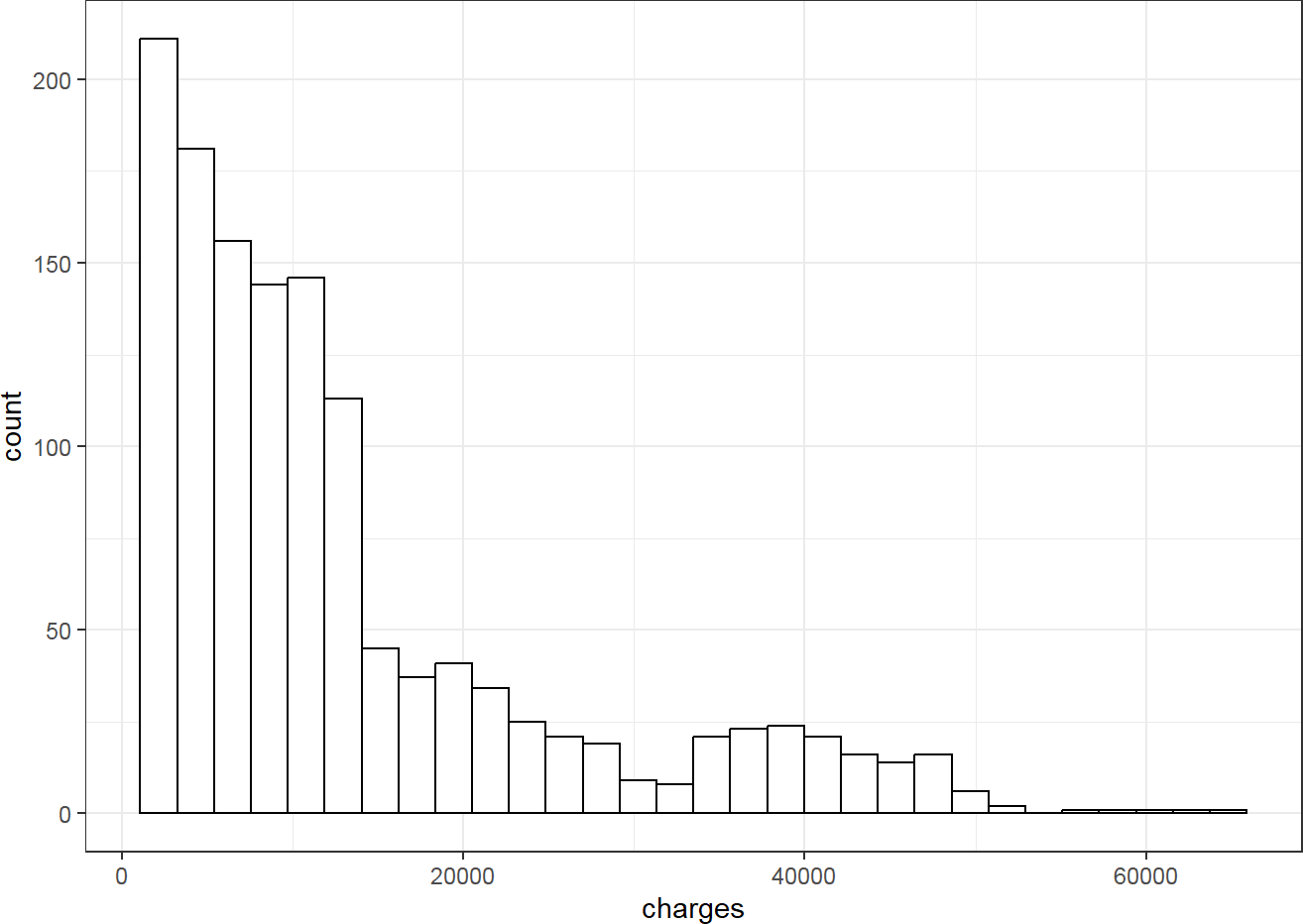
Source: Figure by author(s).

**Figure 40**. Histogram of continuous variable.



Source: Figure by author(s).

**Figure 41**. Histogram of continuous variable.



Source: Figure by author(s).

**Figure 42**. Histogram of continuous variable.

We fit an example simple linear model with “charges” as outcome and “smoker” as categorical predictor.

flm<-lm(charges ~ smoker, data = dlm)

sumflm<-summary(flm)

sumflm

##

## Call:

## lm(formula = charges ~ smoker, data = dlm) ##

## Residuals:

## Min 1Q Median 3Q Max

## -19221 -5042 -919 3705 31720

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 8434.3 229.0 36.83 <0.0000000000000002 \*\*\*

## smokeryes 23616.0 506.1 46.66 <0.0000000000000002 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 7470 on 1336 degrees of freedom

## Multiple R-squared: 0.6198, Adjusted R-squared: 0.6195

## F-statistic: 2178 on 1 and 1336 DF,

## p-value: < 0.00000000000000022

*# Add confidence interval to summary*

ciflm <- confint(flm)

sumciflm <- cbind(sumflm$coefficients, ciflm)

round(sumciflm, 4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | t value | Pr(>|t|) | 2.5 % |
| ## (Intercept) | 8434.268 | 229.0142 | 36.8286 | 0 | 7985.002 |
| ## smokeryes | 23615.964 | 506.0753 | 46.6649 | 0 | 22623.175 |
| ## | 97.5 % |  | | | |
| ## (Intercept) | 8883.535 |
| ## smokeryes | 24608.752 |

From the output, “smoker” = “no” is expected to have charges of 8434.2682979 and “smoker” = “yes” is expected to have charges of 23615.9635337 higher than “smoker” = “no”.

Predicted “charges” for smokers and non-smokers may be calculated and visualized as below using *predict()* and *ggplot()*.

predictions <- predict(flm, newdata = dlm, interval = "confidence")

new\_data <- cbind(dlm, as.data.frame(predictions))

*#plot*

ggplot(dlm, aes(smoker, charges)) +

geom\_point(color = "grey32", size = 0.75) +

geom\_errorbar(data = new\_data, aes(x = smoker,

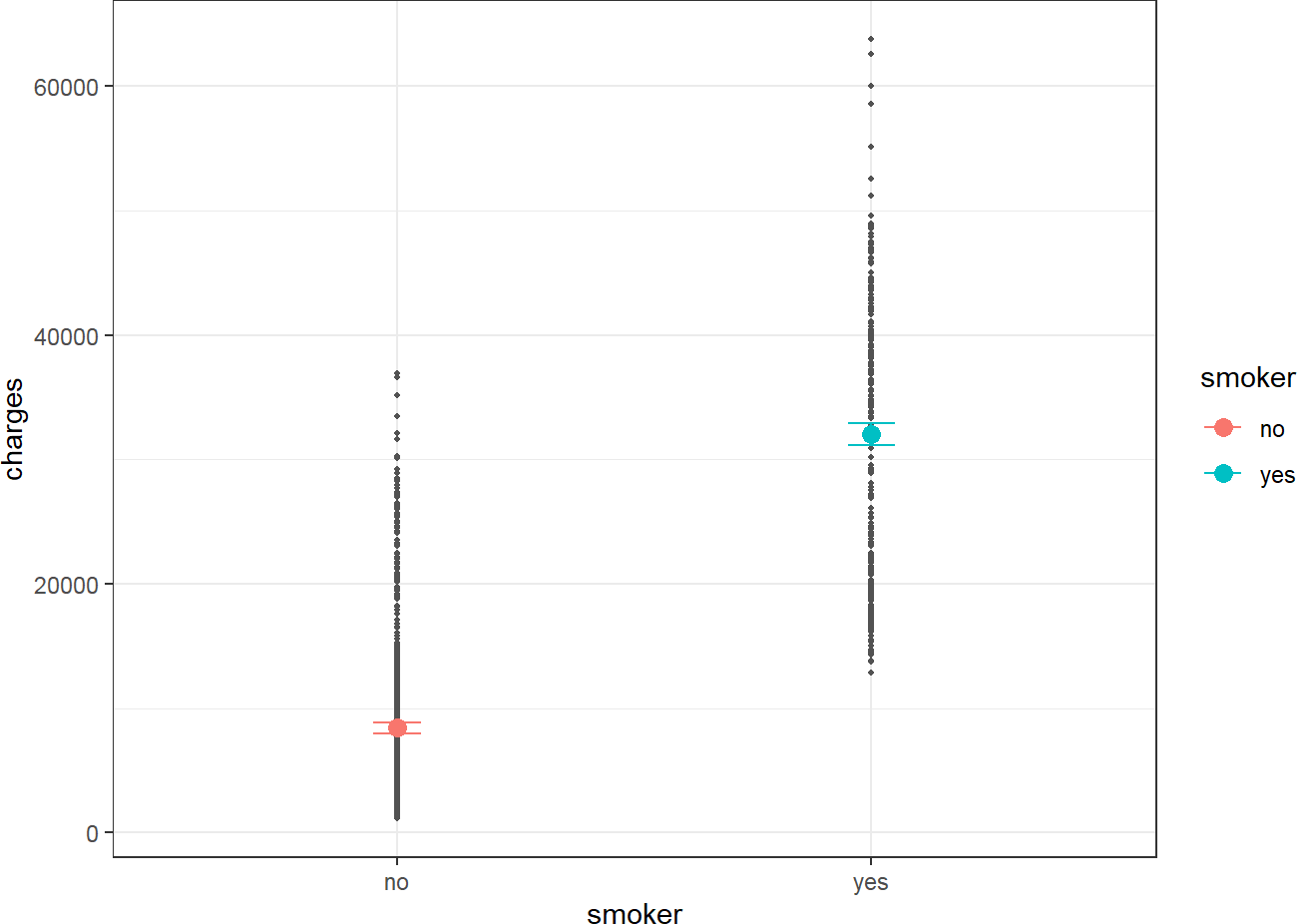
ymin = lwr, ymax = upr, color = smoker),

width = 0.1) +

geom\_point(data = new\_data,

aes(smoker, fit, color = smoker), size = 3) +

labs(x = "smoker", y = "charges") + theme\_bw()



Source: Figure by author(s).

**Figure 43**. Predicted values by smoker group.

The plot shows clear difference in predicted “charges” between non-smoker and smoker corresponding to significant coefficient of “smoker” in the simple linear model.

We fit another example simple linear model with “charges “as outcome and

“region” as categorical predictor (region has 4 categories).

flm.r<-lm(charges ~ region, data = dlm)

sumflm.r<-summary(flm.r)

sumflm.r

##

## Call:

## lm(formula = charges ~ region, data = dlm) ##

## Residuals:

## Min 1Q Median 3Q Max

## -13614 -8463 -3793 3385 49035

##

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ##  ## | Coefficients: | Estimate | Std. | Error | t value | Pr(>|t|) |
| ## | (Intercept) | 13406.4 |  | 671.3 | 19.971 | <0.0000000000000002 |
| ## | regionnorthwest | -988.8 |  | 948.6 | -1.042 | 0.297 |
| ## | regionsoutheast | 1329.0 |  | 922.9 | 1.440 | 0.150 |
| ## | regionsouthwest | -1059.4 |  | 948.6 | -1.117 | 0.264 |
| ## |  |  |  |  |  |  |
| ## | (Intercept) \*\*\* | | | | | |
| ## | regionnorthwest | | | | | |
| ## | regionsoutheast | | | | | |
| ## | regionsouthwest | | | | | |
| ## | --- | | | | | |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 | | | | | |
| ## | Residual standard error: 12080 on 1334 degrees of freedom | | | | | |

## Multiple R-squared: 0.006634, Adjusted R-squared: 0.0044 ## F-statistic: 2.97 on 3 and 1334 DF, p-value: 0.03089

*# Add confidence interval to summary*

ciflm.r <- confint(flm.r)

sumciflm.r <- cbind(sumflm.r$coefficients, ciflm.r )

round(sumciflm.r, 4)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | t value | Pr(>|t|) |
| ## (Intercept) | 13406.3845 | 671.2966 | 19.9709 | 0.0000 |
| ## regionnorthwest | -988.8091 | 948.6263 | -1.0424 | 0.2974 |
| ## regionsoutheast | 1329.0269 | 922.9071 | 1.4400 | 0.1501 |
| ## regionsouthwest | -1059.4471 | 948.6263 | -1.1168 | 0.2643 |
| ## | 2.5 % | 97.5 % |  |  |
| ## (Intercept) | 12089.4724 | 14723.2966 |  |  |
| ## regionnorthwest | -2849.7709 | 872.1526 |  |  |
| ## regionsoutheast | -481.4805 | 3139.5343 |  |  |
| ## regionsouthwest | -2920.4089 | 801.5146 |  |  |

“region” = “northest” is used as reference group and other “region” categories are turned into 3 dummy variables and are compared with the reference group “northest”. From the output, region “northest” is predicted to have “charges” of 13406.3845164. Region “northwest” is predicted to have charges of 988.8091424 lower than region “northest”. Region “southeast” is predicted to have “charges” of 1329.0269212 higher than region “northest”, and so on. However, the difference between “region” categories are not significant.

Predicted “charges” for “region” may be calculated and visualized as below.

predictions <- predict(flm.r, newdata = dlm, interval = "confidence")

new\_data <- cbind(dlm, as.data.frame(predictions))

*#plot*

ggplot(new\_data, aes(region, charges)) +

geom\_point(color = "grey32", size = 0.75) +

geom\_errorbar(data = new\_data, aes(x = region,

ymin = lwr, ymax = upr, color = region),

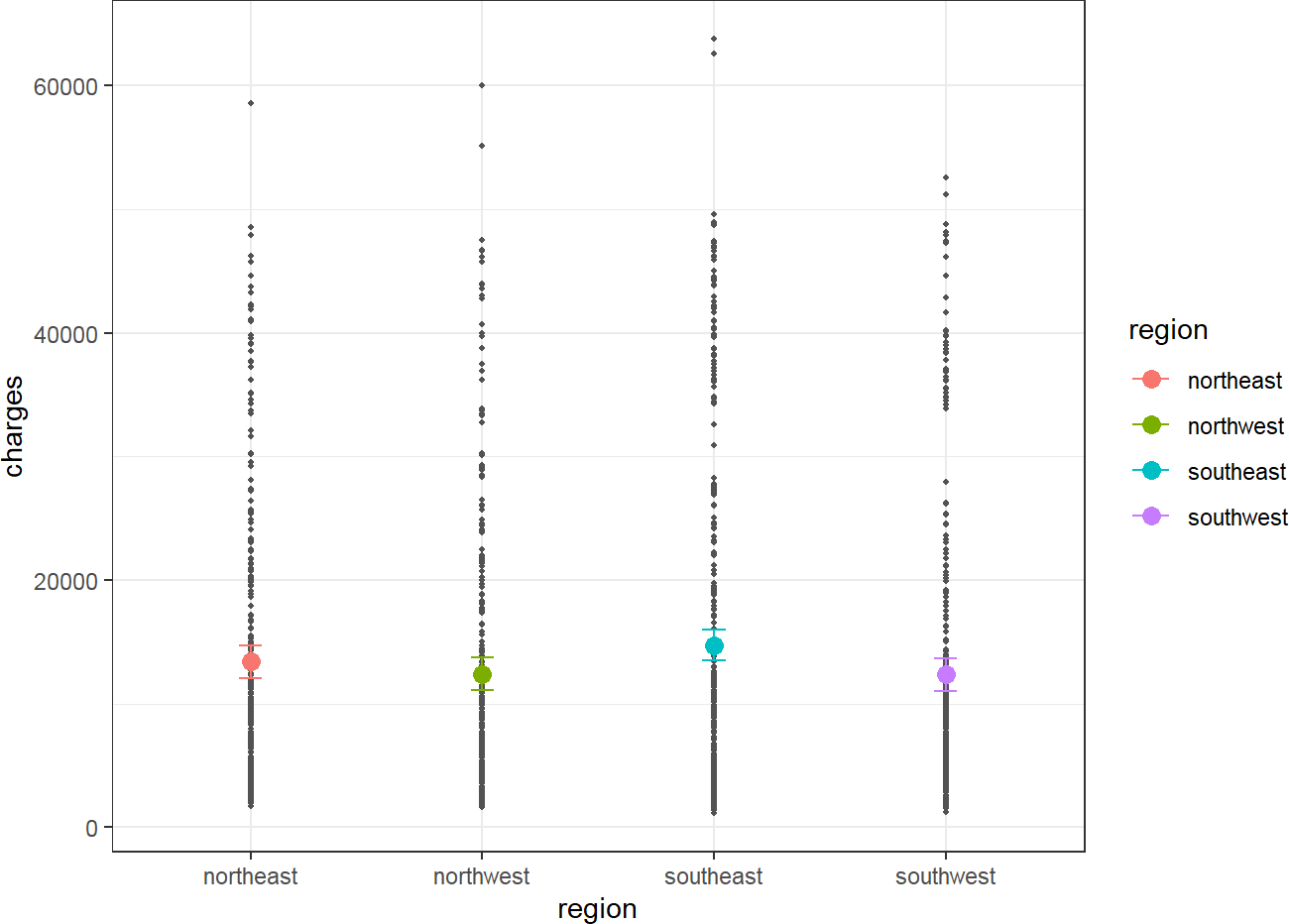
width = 0.1) +

geom\_point(data = new\_data,

aes(region, fit, color = region), size = 3) +

labs(x = "region", y = "charges") + theme(axis.text.x = element\_blank())+

theme\_bw()



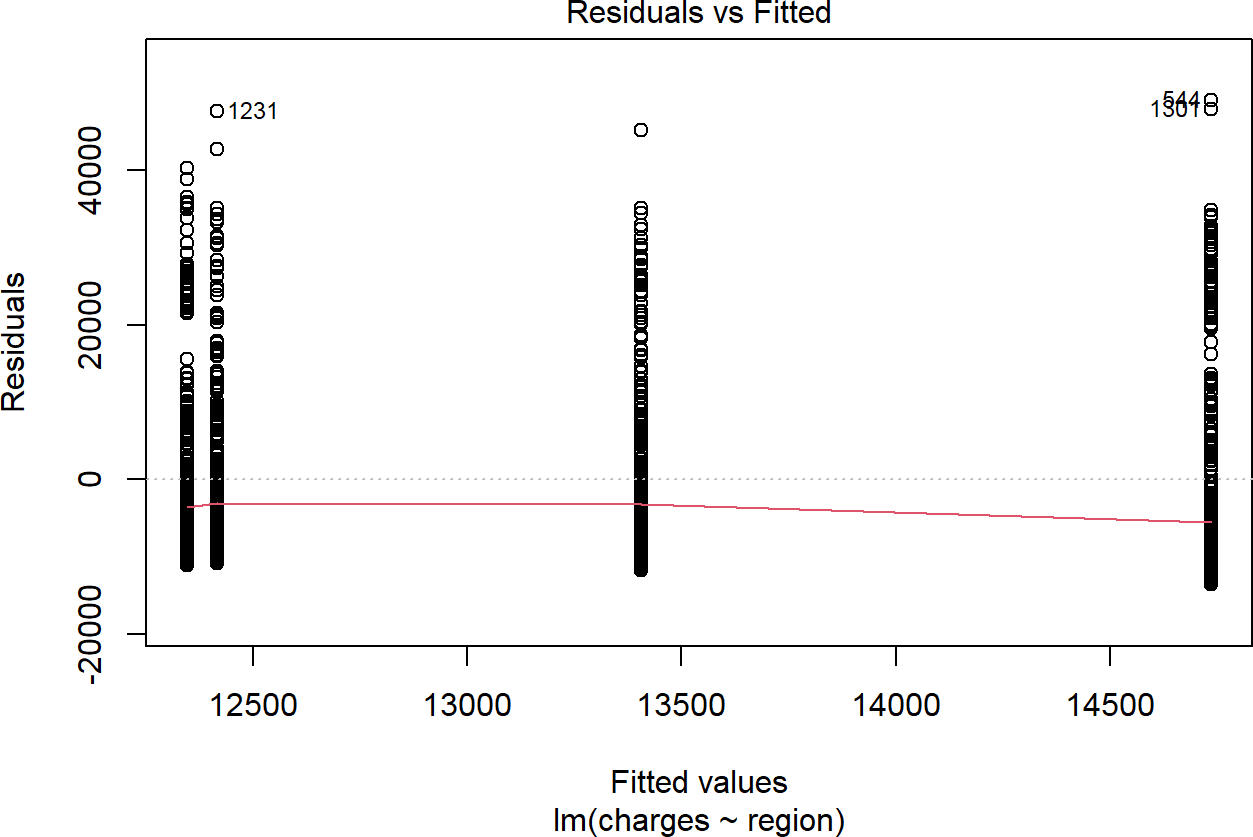
Source: Figure by author(s).

**Figure 44**. Predicted values by region.

The plot shows similar predicted “charges” between “region” categories, which is corresponding to non-significant coefficients of “region” categories vs. reference category “northeast”.

Diagnostic plots may be done with residual vs. fitted plot for homoscedasticity and Q-Q plot for normality of residuals:

plot(flm.r, which = 1)

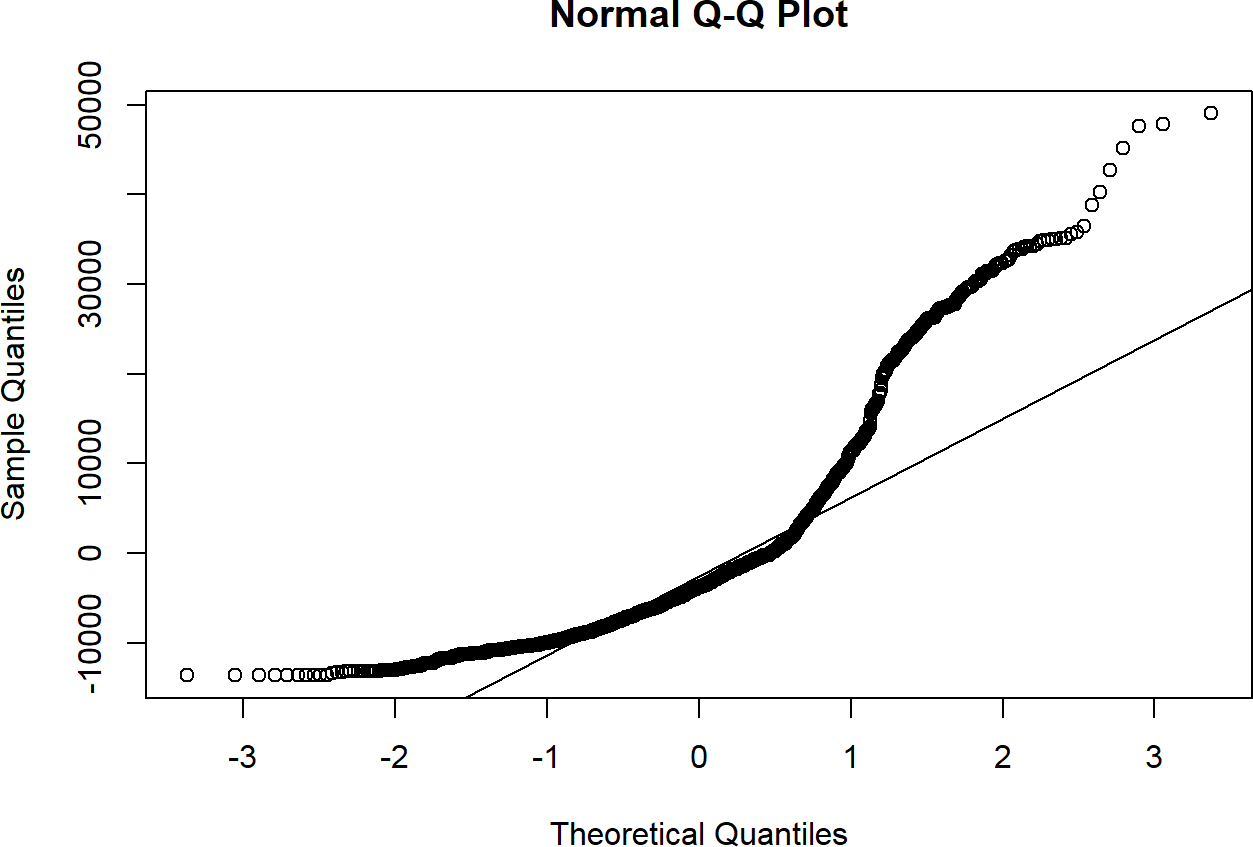


Source: Figure by author(s).

**Figure 45**. Residual vs. fitted plot 2.

qqnorm(residuals(flm.r))

qqline(residuals(flm.r))



Source: Figure by author(s).

**Figure 46**. Q-Q plot 2.

With the above Q-Q plot, the points form a strong S-curve and deviate significantly from the diagonal line, thus, normality assumption seems not met. The histogram of “charges” also shows skewed distribution (not symmetric bell shape). Therefore, you may try logarithm transformation of the outcome “charges”, refit linear model and re-perform diagnostic plots.

dlm$logcharges<-log(dlm$charges)

flm.lr<-lm(logcharges ~ region, data = dlm)

sumflm.lr<-summary(flm.lr)

sumflm.lr

##

## Call:

## lm(formula = logcharges ~ region, data = dlm) ##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | Residuals: |  | | | |
| ## | Min | 1Q Median | 3Q | Max | |
| ## | -2.09965 -0.64192 0.03622 0.62333 | | | 1.94064 |  |
| ## |  | | |  |  |
| ## | Coefficients: | | |  |  |
| ## | Estimate Std. Error | | | t value | Pr(>|t|) |
| ## | (Intercept) 9.16877 0.05106 | | | 179.562 | <0.0000000000000002 |
| ## | regionnorthwest -0.09903 0.07216 | | | -1.372 | 0.1701 |
| ## | regionsoutheast -0.04637 0.07020 | | | -0.660 | 0.5091 |
| ## | regionsouthwest -0.13767 0.07216 | | | -1.908 | 0.0566 |
| ## |  | | |  |  |
| ## | (Intercept) \*\*\* | | |  |  |
| ## | regionnorthwest | | |  |  |
| ## | regionsoutheast | | |  |  |
| ## | regionsouthwest . | | |  |  |
| ## | --- | | |  |  |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' | | | 0.01 '\*' | 0.05 '.' 0.1 ' ' 1 |

## Residual standard error: 0.9191 on 1334 degrees of freedom

## Multiple R-squared: 0.003143, Adjusted R-squared: 0.0009012 ## F-statistic: 1.402 on 3 and 1334 DF, p-value: 0.2406

*# Add confidence interval to summary*

ciflm.lr <- confint(flm.lr)

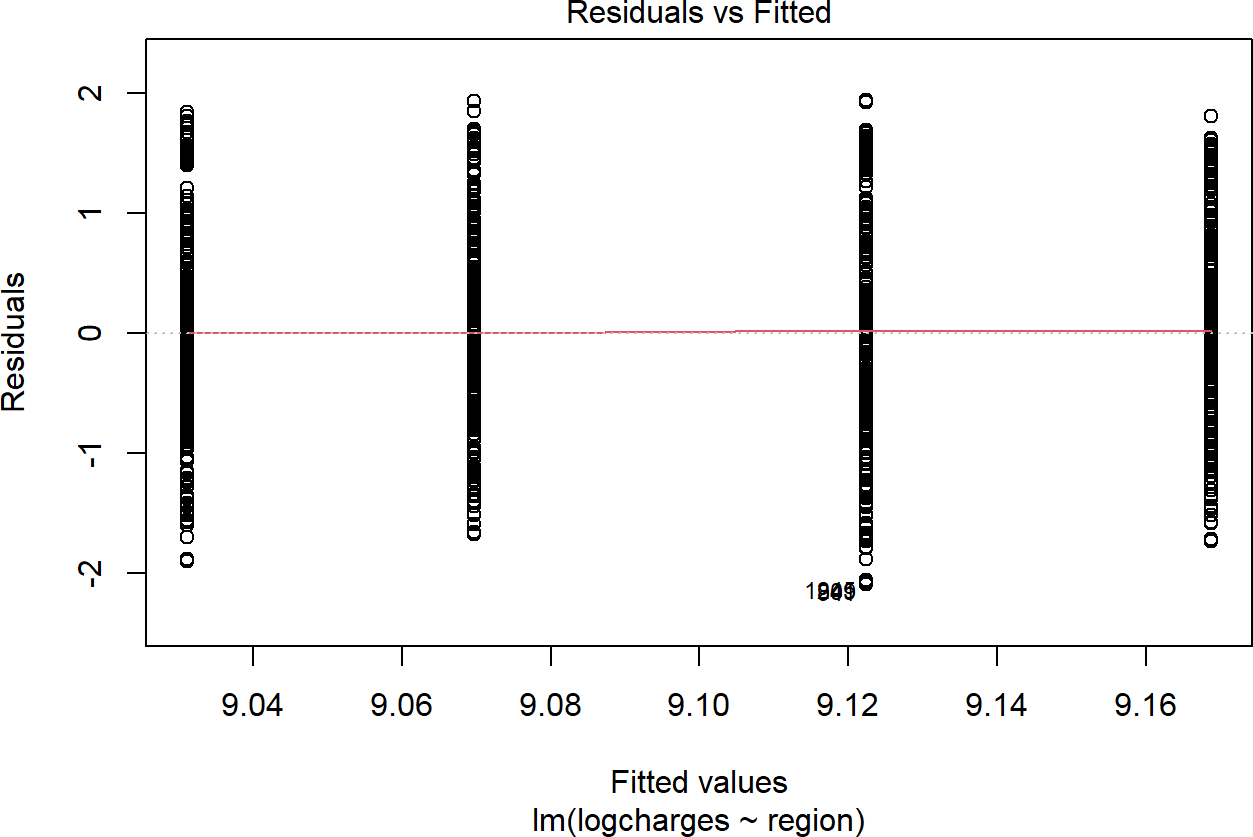
sumciflm.lr <- cbind(sumflm.lr$coefficients, ciflm.lr)

round(sumciflm.lr, 4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | t value | Pr(>|t|) | 2.5 % |
| ## (Intercept) | 9.1688 | 0.0511 | 179.5621 | 0.0000 | 9.0686 |
| ## regionnorthwest | -0.0990 | 0.0722 | -1.3725 | 0.1701 | -0.2406 |
| ## regionsoutheast | -0.0464 | 0.0702 | -0.6605 | 0.5091 | -0.1841 |
| ## regionsouthwest | -0.1377 | 0.0722 | -1.9079 | 0.0566 | -0.2792 |
| ## | 97.5 % |  | | | |
| ## (Intercept) | 9.2689 |
| ## regionnorthwest | 0.0425 |
| ## regionsoutheast | 0.0913 |
| ## regionsouthwest | 0.0039 |

*#diagnostic plot*

plot(flm.lr, which = 1)

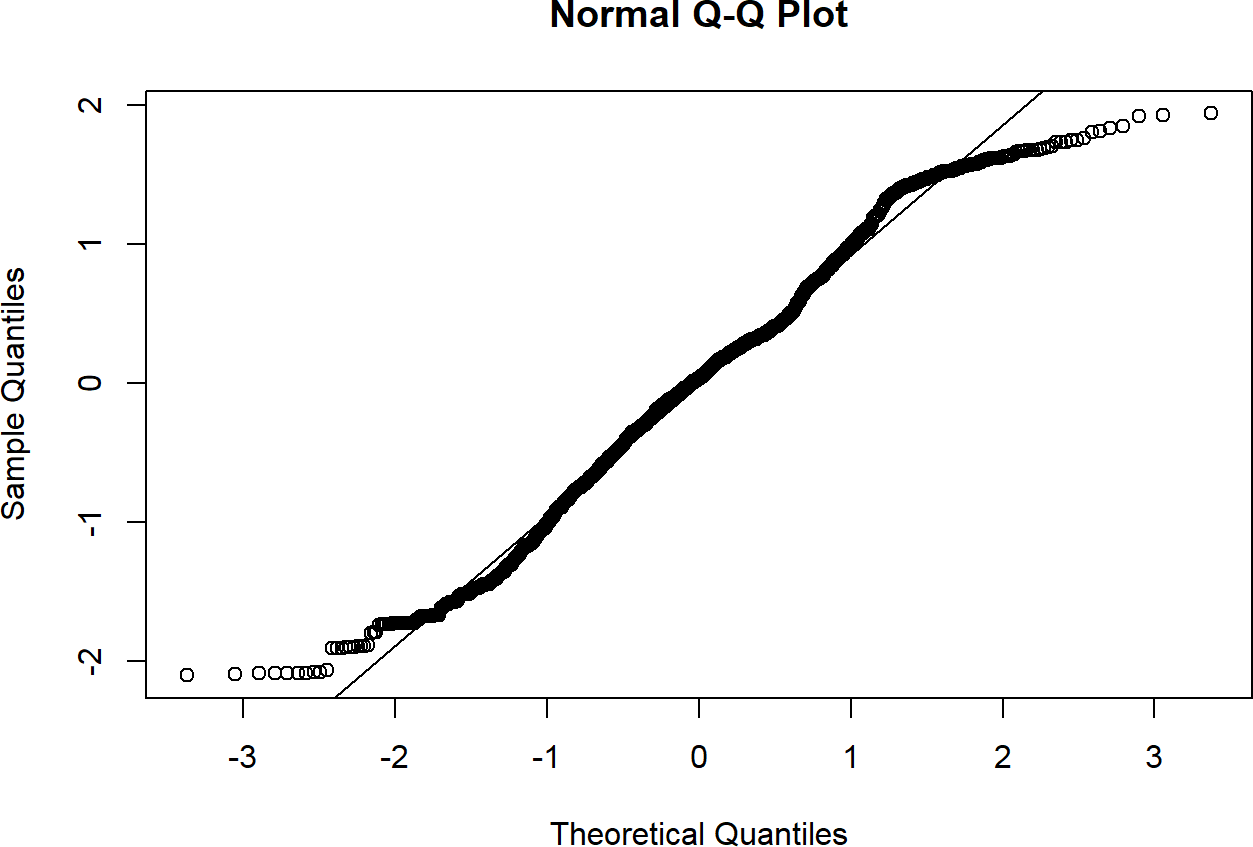


Source: Figure by author(s).

**Figure 47**. Residual vs. fitted plot: log tranformation.

qqnorm(residuals(flm.lr))

qqline(residuals(flm.lr))



Source: Figure by author(s).

**Figure 48**. Q-Q plot: log transformation.

The diagnostic plots look better after logarithm transformation of the outcome

“charges”. Residual vs. fitted plot shows that the vertical spread (variance) within each of the four groups is now much more similar, indicating that the log transformation has stabilized the variance and largely fixed the heteroscedasticity. The Q-Q plot shows that the points now fall much closer to the diagonal line, indicating the residuals of this new model are approximately normally distributed. The log-transformation was a necessary and successful step to meet the model's assumptions. All subsequent models will use “logcharges” as the outcome.

## Multiple Linear Regression

Multiple linear regression allows the estimation of the effects of multiple predictor variables simultaneously on outcome variable. It allows us to estimate the effect of a variable while controlling for other variables in the model. For the example medical costs and patient information data above, multiple linear regression may be done by adding several variables as predictors in the linear model, for example “bmi” and “smoker” as predictors and logarithm of “charges” (“logcharges”) as outcome.

*# Simple linear regression model with only bmi as*

*# predictor and logarithm charges as outcome*

flm.b<-lm(logcharges ~ bmi, data = dlm)

sumflm.b<-summary(flm.b)

sumflm.b

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## |  | | | | | |
| ## | Call: | | | | |  |
| ## | lm(formula = logcharges ~ bmi, data = dlm) | | | | |  |
| ## |  | | | | |  |
| ## | Residuals: | | | | |  |
| ## | Min 1Q Median 3Q Max | | | | |  |
| ## | -2.48894 -0.63536 0.03136 0.68007 1.95182 | | | | |  |
| ## |  | | | | |  |
| ## | Coefficients: | | | | |  |
| ## | Estimate Std. Error t value Pr(>|t|) | | | | |  |
| ## | (Intercept) 8.485243 0.127833 66.378 < 0.0000000000000002 | | | | | \*\*\* |
| ## | bmi | 0.020005 | 0.004089 | 4.892 | 0.00000112 | \*\*\* |
| ## | --- |  |  |  |  |  |
| ## ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 | | | | '\*' 0.05 '.' 0.1 ' ' | 1 |

## Residual standard error: 0.9117 on 1336 degrees of freedom

## Multiple R-squared: 0.0176, Adjusted R-squared: 0.01687 ## F-statistic: 23.94 on 1 and 1336 DF, p-value: 0.000001117

*# Add confidence interval to summary*

ciflm.b <- confint(flm.b)

sumciflm.b <- cbind(sumflm.b$coefficients, ciflm.b)

round(sumciflm.b, 4)

## Estimate Std. Error t value Pr(>|t|) 2.5 % 97.5 %

## (Intercept) 8.4852 0.1278 66.3778 0 8.2345 8.736

## bmi 0.0200 0.0041 4.8925 0 0.0120 0.028

*# Simple linear regression model with only smoker as predictor*

*# and logarithm charges as outcome*

flm.s<-lm(logcharges ~ smoker, data = dlm)

sumflm.s<-summary(flm.s)

sumflm.s

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## |  | | | | | |
| ## | Call: | | | | |  |
| ## | lm(formula = logcharges ~ smoker, data = dlm) | | | | |  |
| ## |  | | | | |  |
| ## | Residuals: | | | | |  |
| ## | Min 1Q Median 3Q Max | | | | |  |
| ## | -1.7655 -0.4370 0.1237 0.4785 1.7280 | | | | |  |
| ## |  | | | | |  |
| ## | Coefficients: | | | | |  |
| ## | Estimate Std. Error t value Pr(>|t|) | | | | |  |
| ## | (Intercept) 8.78823 0.02105 417.52 <0.0000000000000002 | | | | | \*\*\* |
| ## | smokeryes | 1.51588 | 0.04651 | 32.59 | <0.0000000000000002 | \*\*\* |
| ## | --- |  |  |  |  |  |
| ## ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 | | | | | |
| ## | Residual standard error: 0.6866 on 1336 degrees of freedom | | | | | |
| ## | Multiple R-squared: 0.4429, Adjusted R-squared: 0.4425 | | | | | |
| ## | F-statistic: 1062 on 1 and 1336 DF, p-value: < 0.00000000000000022 | | | | | |

*# Add confidence interval to summary*

ciflm.s <- confint(flm.s)

sumciflm.s <- cbind(sumflm.s$coefficients, ciflm.s)

round(sumciflm.s, 4)

## Estimate Std. Error t value Pr(>|t|) 2.5 % 97.5 % ## (Intercept) 8.7882 0.0210 417.5206 0 8.7469 8.8295

## smokeryes 1.5159 0.0465 32.5902 0 1.4246 1.6071

*# Multiple linear regression model with smoker and*

*# bmi as predictors and logarithm charges as outcome*

flm.sb<-lm(logcharges ~ smoker + bmi, data = dlm)

sumflm.sb<-summary(flm.sb)

sumflm.sb

##

## Call:

## lm(formula = logcharges ~ smoker + bmi, data = dlm) ##

## Residuals:

## Min 1Q Median 3Q Max

## -2.17030 -0.40754 0.09871 0.44368 1.75504

##

## Coefficients:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | t value | Pr(>|t|) |  |
| ## (Intercept) | 8.186576 | 0.095254 | 85.945 < | 0.0000000000000002 | \*\*\* |
| ## smokeryes | 1.514765 | 0.045818 | 33.061 < | 0.0000000000000002 | \*\*\* |
| ## bmi | 0.019629 | 0.003033 | 6.472 | 0.000000000136 | \*\*\* |
| ## --- |  |  |  |  |  |
| ## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' '  ## | | | | | 1 |
| ## Residual standard error: 0.6763 on 1335 degrees of freedom | | | | |  |
| ## Multiple R-squared: 0.4598, Adjusted R-squared: 0.459 | | | | |  |
| ## F-statistic: 568.3 on 2 and 1335 DF, p-value: < 0.00000000000000022 | | | | |  |

*# Add confidence interval to summary*

ciflm.sb <- confint(flm.sb)

sumciflm.sb <- cbind(sumflm.sb$coefficients, ciflm.sb)

round(sumciflm.sb, 4)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | Estimate | | Std. Error | t value | Pr(>|t|) | 2.5 % | 97.5 | % |
| ## | (Intercept) | 8.1866 | 0.0953 | 85.9449 | 0 | 7.9997 | 8.3734 | |
| ## | smokeryes | 1.5148 | 0.0458 | 33.0606 | 0 | 1.4249 | 1.6046 | |
| ## | bmi | 0.0196 | 0.0030 | 6.4715 | 0 | 0.0137 | 0.0256 | |

From the output of multiple linear regression, the coefficients change slightly for both “smoker” and “bmi” as compared to those of the simple linear regression models when one of these predictors was used alone.

The interpretation of coefficients is as below:

* “logcharges” when “smoker” = no and “bmi” *= 0* is 8.1865764
* After controlling for “bmi”, “smoker” = yes is expected to have “logcharges” of 1.514765 higher than “smoker” = no.
* After controlling for “smoker”, each unit increase in “bmi” is associated with an increase of 0.0196287 in “logcharges”.

Both *smokeryes* and *bmi* have very small p-values, so both are significant predictors even when controlling for each other. R-squared of 0.4598 indicates that together, these two variables explain 45.98% of the variance in “logcharges”.

Observed and predicted values with 95% confidence intervals of the above multiple regression model may be visualized as below:

predictions <- predict(flm.sb, newdata = dlm, interval = "confidence")

new\_data <- cbind(dlm, as.data.frame(predictions))

*#plot*

ggplot(new\_data, aes(bmi, logcharges, color = smoker)) +

geom\_point() +

geom\_ribbon(data = new\_data, aes(x = bmi, ymin = lwr, ymax = upr, fill = smoker),

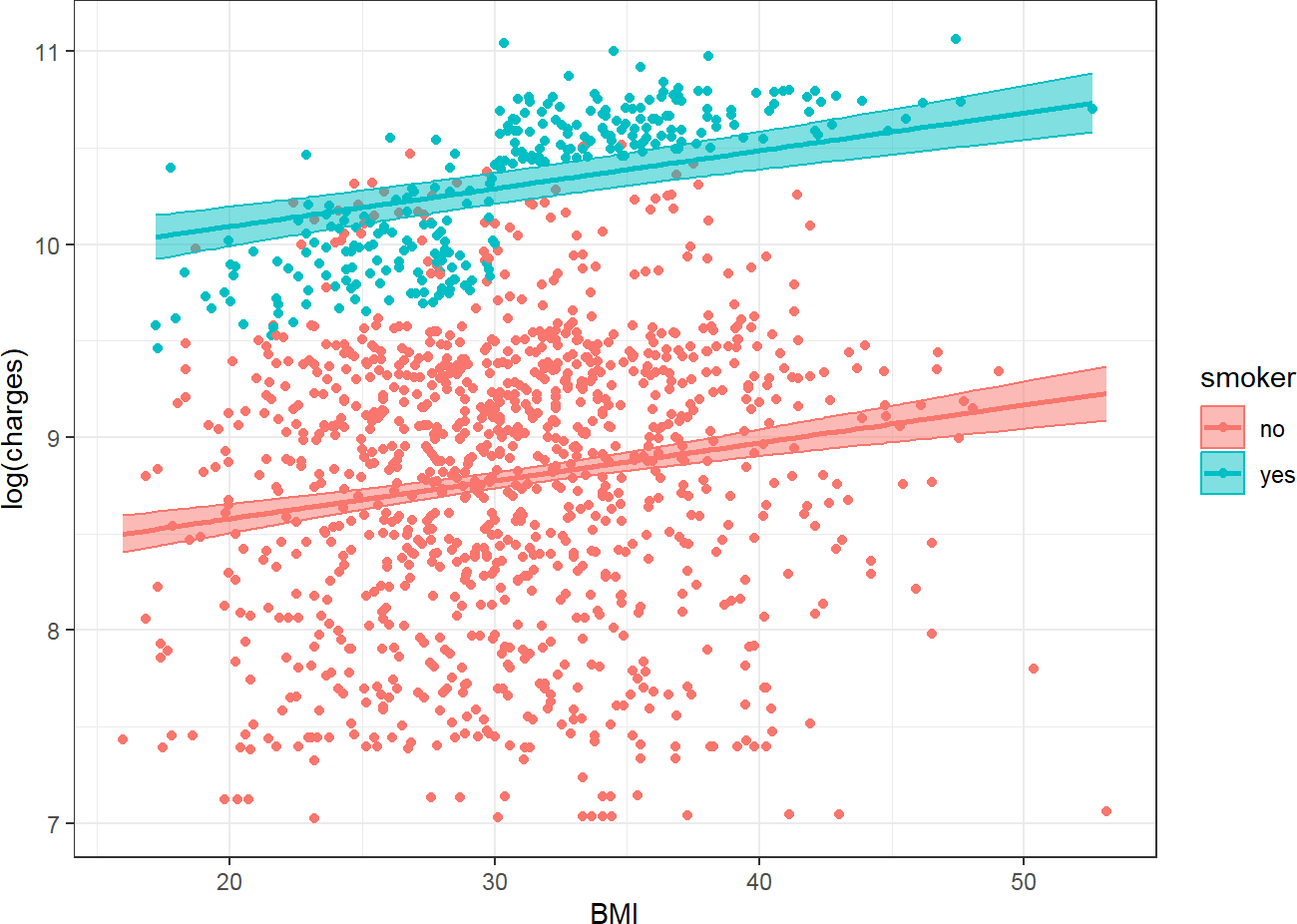
alpha = 0.5) +

geom\_line(data = new\_data, aes(bmi, fit, color = smoker),

linewidth = 1) +

labs(x = "BMI", y = "log(charges)") +

theme\_bw()



Source: Figure by author(s).

**Figure 49**. Observed and predicted values with 95%CI

In the above multiple linear regression model with “smoker” and “bmi” as predictors, the slopes for “logcharges” against “bmi” is the same for “smoker” = no and “smoker” = yes. However, the above plot shows the difference in the association of “logcharges” against “bmi” between “smoker” = no and “smoker” = yes. Thus, the multiple linear regression model above may not fit the data well, especially for “smoker”= yes.

To address this difference, an interaction term between “smoker” and “bmi” may be added to the model:

flm.i<-lm(logcharges ~ smoker + bmi +smoker\*bmi, data = dlm)

sumflm.i<-summary(flm.i)

sumflm.i

##

## Call:

## lm(formula = logcharges ~ smoker + bmi + smoker \* bmi, data = dlm)

##

## Residuals:

## Min 1Q Median 3Q Max

## -1.98216 -0.37399 0.07215 0.44881 1.72280

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | (Intercept) | 8.443124 | 0.106184 | 79.514 | < | 0.0000000000000002 |
| ## | smokeryes | 0.342811 | 0.227399 | 1.508 |  | 0.131911 |
| ## | bmi | 0.011259 | 0.003399 | 3.313 |  | 0.000949 |
| ## | smokeryes:bmi | 0.038179 | 0.007259 | 5.259 |  | 0.000000168 |
| ## |  |  |  | | | |
| ## | (Intercept) | \*\*\* |
| ## | smokeryes |  |
| ## | bmi | \*\*\* |
| ## | smokeryes:bmi | \*\*\* |
| ## | --- |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' | ' | 1 |
| ##  ## | Residual standard error: 0.6697 on 1334 degrees of freedom |  |  |
| ##  ## | Multiple R-squared: 0.4708, Adjusted R-squared: 0.4696  F-statistic: 395.6 on 3 and 1334 DF, p-value: < 0.00000000000000022 |  |  |

*# Add confidence interval to summary*

ciflm.i <- confint(flm.i)

sumciflm.i <- cbind(sumflm.i$coefficients, ciflm.i) round(sumciflm.i, 4)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | t value | Pr(>|t|) | 2.5 % |
| ## (Intercept) | 8.4431 | 0.1062 | 79.5145 | 0.0000 | 8.2348 |
| ## smokeryes | 0.3428 | 0.2274 | 1.5075 | 0.1319 | -0.1033 |
| ## bmi | 0.0113 | 0.0034 | 3.3126 | 0.0009 | 0.0046 |
| ## smokeryes:bmi | 0.0382 | 0.0073 | 5.2595 | 0.0000 | 0.0239 |
| ## | 97.5 % |  | | | |
| ## (Intercept) | 8.6514 |
| ## smokeryes | 0.7889 |
| ## bmi | 0.0179 |
| ## smokeryes:bmi | 0.0524 |

And we re-perform the plot of the model with interaction term:

predictions <- predict(flm.i, newdata = dlm, interval = "confidence")

new\_data <- cbind(dlm, as.data.frame(predictions))

*#plot*

ggplot(new\_data, aes(bmi, logcharges, color = smoker)) +

geom\_point() +

geom\_ribbon(data = new\_data, aes(x = bmi, ymin = lwr,

ymax = upr, fill = smoker),

alpha = 0.5) +

geom\_line(data = new\_data, aes(bmi, fit, color = smoker), linewidth = 1) +

labs(x = "BMI", y = "log(charges)") + theme\_bw()



Source: Figure by author(s).

**Figure 50**. Observed and predicted values with interaction.

Now the plot shows two different slopes for “smoker” = no and “smoker” = yes and the model fits the data better especially for “smoker” = yes.

The interpretation of coefficients of the multiple linear regression model with interaction term is as below:

* “logcharges” when “smoker” = no and “bmi” = 0 is 8.443124
* When “bmi” = 0, “logcharges” when “smoker” = yes is 0.3428113 higher when “smoker” = no
* When “smoker” = no, for each unit increase in “bmi”, “logcharges” increases by 0.011259
* Coefficient for interaction term expresses the change in a variable’s effect when

the interacting variable is increased by one-unit:

* + The effect of “bmi” on “logcharges” increases by 0.0381793 (“logcharges” per one unit of “bmi”) in “smoker” = yes vs. “smoker” = no.
  + For each unit increase in “bmi”, the effect of “smoker” = yes vs. “smoker” = no on “logcharges” increases by 0.0381793.

The effect of “bmi” for “smoker” = yes or “smoker” = no may be calculated by adding the coefficient of “bmi” 0.011259 to the interaction coefficient, 0.0381793, multiplied by “smoker” as dummy variable 0, 1:

* “bmi” effect (“smoker” = no) = 0.011259 + 0.0381793 \*0
* “bmi” effect (“smoker” = yes) = 0.011259 + 0.0381793 \*1

The effect of “smoker” for any “bmi” may be calculated by adding the coefficient of “smoker” 0.3428113 to the interaction coefficient, 0.0381793 and multiplied by “bmi”:

* “smoker” effect (“bmi” = 15) = 0.3428113 + 0.0381793 \*15

## Model Selection

Model selection is the process of systematically choosing the best subset of predictors from a larger set to include in the final model. The goal is to create a model that is both parsimonious (as simple as possible) and has high explanatory power. Several statistical criteria may be used for model selection:

* Adjusted R-squared: penalizes the model for having more predictors. It increases only if the new predictor improves the model more than would be expected by chance. When comparing models, a model with higher adjusted R-squared is usually preferred.
* Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC):

These are information-based criteria that balance model fit with model complexity. Lower values indicate a better model.

Below is an example using *step()* function to perform both backward and forward stepwise selection based on AIC. First, a *full\_model* is created containing all predictors we want to consider. The *step(full\_model, ...)* function then iteratively adds and removes predictors from this model, checking the AIC at each step (*trace = 0* simply suppresses the printing of every single step). The final result, *stepwise\_model*, is the model with the lowest possible AIC that the step function could find.

*# Fit a full model with multiple predictors*

full\_model <- lm(logcharges ~ age + sex + bmi + smoker +

smoker\*bmi + children + region, data = dlm)

*# Use the step() function to perform backward*

*# stepwise selection based on AIC*

*# The 'direction' can be "backward", "forward", or "both"*

stepwise\_model <- step(full\_model, direction = "both", trace = 0)

*# View the summary of the selected model*

summary(stepwise\_model)

##

## Call:

## lm(formula = logcharges ~ age + sex + bmi + smoker + smoker \*

## bmi + children + region, data = dlm)

##

## Residuals:

## Min 1Q Median 3Q Max

## -0.91507 -0.17764 -0.05143 0.04641 2.22317

##

## Coefficients:

## Estimate Std. Error t value

## (Intercept) 7.3373697 0.0766733 95.697

## age 0.0347953 0.0008429 41.281

## sexmale -0.0870645 0.0236073 -3.688

## bmi 0.0034060 0.0022676 1.502

## smokeryes 0.1564189 0.1459995 1.071

## children 0.1031486 0.0097594 10.569

## regionnorthwest -0.0711306 0.0337354 -2.108

## regionsoutheast -0.1627269 0.0339029 -4.800

## regionsouthwest -0.1375125 0.0338557 -4.062

## bmi:smokeryes 0.0455744 0.0046633 9.773 ## Pr(>|t|)

## (Intercept) < 0.0000000000000002 \*\*\*

## age < 0.0000000000000002 \*\*\*

## sexmale 0.000235 \*\*\*

## bmi 0.133336

## smokeryes 0.284199

## children < 0.0000000000000002 \*\*\*

## regionnorthwest 0.035176 \*

## regionsoutheast 0.00000177 \*\*\*

## regionsouthwest 0.00005156 \*\*\*

## bmi:smokeryes < 0.0000000000000002 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.4293 on 1328 degrees of freedom

## Multiple R-squared: 0.7835, Adjusted R-squared: 0.7821

## F-statistic: 534 on 9 and 1328 DF, p-value:<0.00000000000000022

The *summary(stepwise\_model)* output shows the final model chosen by both forward and backward selection algorithm. In this example, all variables in the full model are retained in the final model.

## Model Validation

Model evaluation and selection focus on how well the model fits the data it was built on (training data). Model validation examines the performance of a model on new (testing) data. The most common method is to split the dataset into two parts:

* Training Set: A subset of the data (e.g., 80%) used to build and train the model.
* Testing Set: The remaining portion of the data (e.g., 20%) that is held back and used only to evaluate the performance of the trained model.

### Performance Metrics

* + - * Mean Absolute Error (MAE): the average of the absolute differences between the predicted and actual values.
      * Mean Squared Error (MSE): the average of the squared prediction errors.
      * Root Mean Squared Error (RMSE): the square root of the MSE (the most

commonly used regression metric).

* + - * R-squared (*R*2): the proportion of the variance in the test set’s outcome that is

explained by the model. A significant drop in R-squared from the training set to the test set is a strong indicator of overfitting.

Below are examples to fit the model on training data, calculate the above performance metrics on testing data using the “*yardstick”* package [33]. First, split the data, fit the model on training data and predict on testing data:

*#library(yardstick)*

*# For calculating performance metrics*

*# Split the data (reproducing the split for a self-contained example)*

*# Set a seed for reproducibility*

set.seed(123)

*# Create an index for the training set (80% of the data)*

train\_index <- sample(1:nrow(dlm), 0.8 \* nrow(dlm))

*# Create the training and testing datasets*

train\_data <- dlm[train\_index, ]

test\_data <- dlm[-train\_index, ]

*# Train the model on the training data ONLY*

final\_model <- lm(logcharges ~ age + sex + bmi + smoker +

smoker\*bmi + children + region, data = train\_data)

*# Make predictions and combine with actuals*

*# Create a results data frame with actuals and predictions*

results <- test\_data %>%

mutate(Actual = logcharges) %>%

mutate(Predicted = predict(final\_model, newdata = test\_data))%>%

select(Actual,Predicted )

*# View the first few rows of the results*

head(results)

|  |  |  |  |
| --- | --- | --- | --- |
| ## |  | Actual | Predicted |
| ## | 14 | 9.313864 | 9.262429 |
| ## | 15 | 10.586881 | 10.276976 |
| ## | 21 | 9.490155 | 9.545169 |
| ## | 22 | 8.330800 | 8.475589 |
| ## | 27 | 9.578577 | 9.596659 |
| ## | 33 | 8.452718 | 8.485702 |

Then, calculate performance metrics using “*yardstick”* functions. The “*yardstick”* functions follow a consistent format: *metric\_function(data, truth = Actual\_Column\_Name, estimate = Predicted\_Column\_Name)*.

We calculate Root Mean Squared Error (RMSE) using *rmse()*:

rmse\_value <- rmse(results, truth = Actual, estimate = Predicted)

print(rmse\_value)

## # A tibble: 1 x 3

## .metric .estimator .estimate

## <chr> <chr> <dbl>

## 1 rmse standard 0.429

The output shows that, on average, the model’s prediction of “logcharges” is off by about 0.4294596 on the testing data.

Calculate Mean Absolute Error (MAE) using *mae()*:

mae\_value <- mae(results, truth = Actual, estimate = Predicted)

print(mae\_value)

## # A tibble: 1 x 3

## .metric .estimator .estimate

## <chr> <chr> <dbl>

## 1 mae standard 0.270

The output shows that the average absolute error of any given prediction is 0.2695986.

We calculate R-squared using *rsq()*:

rsq\_value <- rsq(results, truth = Actual, estimate = Predicted)

print(rsq\_value)

## # A tibble: 1 x 3

## .metric .estimator .estimate

## <chr> <chr> <dbl>

## 1 rsq standard 0.793

The output shows that the model explains approximately 0.7928342 of the variance in actual “logcharges” in the testing data. This value is very close to the R-squared from the training data. This similarity is excellent and confirms that our model is not overfit and generalizes well.

### Calibration

Model calibration assesses the agreement between the model’s predictions and the actual observed outcomes. A well-calibrated model is one where the predictions are reliable across the entire range of values. A model may have a low average error (as evaluated by the above performance metrics) but may have poor calibration which makes the model not trust worthy for decision-making at the individual level. In other words, while RMSE and R-squared give an average error, a calibration plot shows where the model is strong and weak.

Calibration is best evaluated by calibration plot:

ggplot(results, aes(x = Predicted, y = Actual)) +

*# Add the scatter plot of points.*

*# Alpha transparency helps visualize data density.*

geom\_point(alpha = 0.3, color = "gray40") +

*# Add the line of perfect calibration (y = x line).*

*# This is the reference line where Predicted would equal Actual.*

geom\_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed", linewidth = 1) +

*# Add a smoothed line to show the average trend of our model's predictions. # This is the "calibration curve" that we will evaluate.*

geom\_smooth(method = "loess", se = FALSE, color = "blue", linewidth = 1.2) +

*# Ensure the plot axes are scaled equally for a proper 45-degree line view*

coord\_fixed(

xlim = range(c(results$Predicted, results$Actual)), ylim = range(c(results$Predicted, results$Actual))

) +

*# Add informative labels and a clean theme*

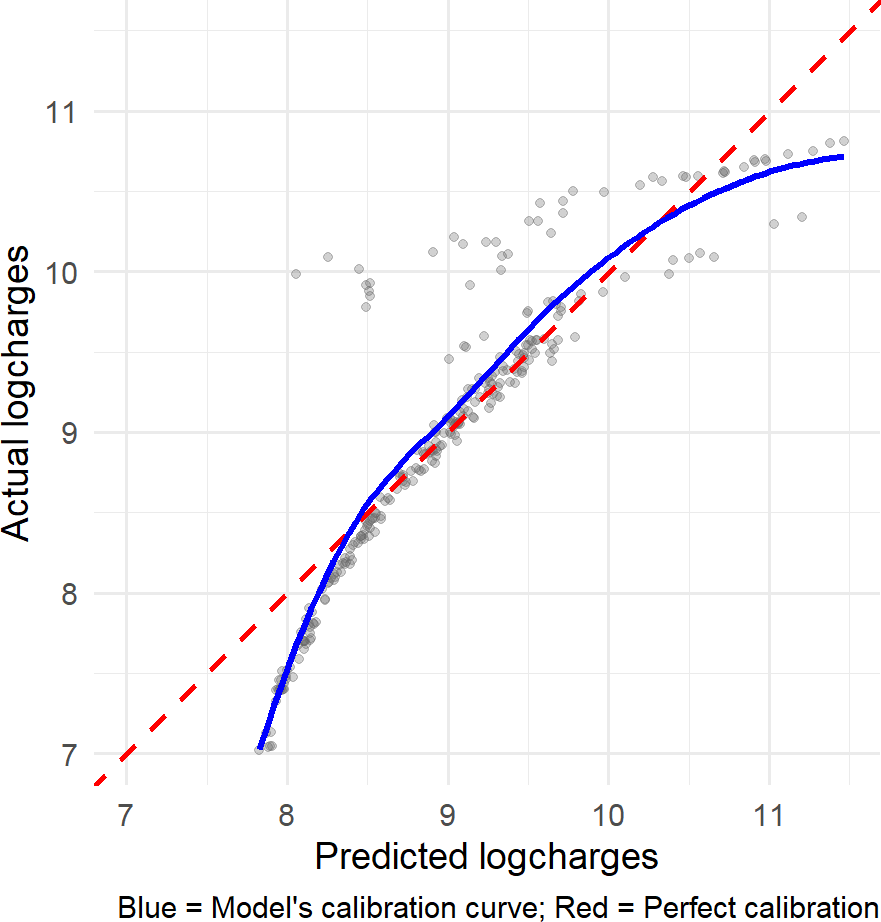
labs(

x = "Predicted logcharges", y = "Actual logcharges",

caption = "Blue = Model's calibration curve; Red = Perfect calibration"

) +

theme\_minimal(base\_size = 14)



Source: Figure by author(s).

**Figure 51**. Calibration plot.

In the calibration plot above, the Predicted values (x-axis) are plotted against the Actual values (y-axis). The red dashed line is the line of perfect calibration, where Predicted = Actual. The blue solid line is the calibration curve, showing the average trend of our model's predictions. A well-calibrated model would have its blue line fall directly on top of the red dashed line.

The plot shows that the model tends to under predict for small and large

“logchages” values, while predict relatively well for middle “logchages” values.

## Useful Resources

* [Introduction to Statistical Learning: with Applications in R](https://www.stat.berkeley.edu/~rabbee/s154/ISLR_First_Printing.pdf) [38].
* [Introduction to Linear Regression in R by UCLA Office of Advanced Research](https://stats.oarc.ucla.edu/r/seminars/intro-to-linear-regression-r/) [Computing Statistical Methods and Data Analytics](https://stats.oarc.ucla.edu/r/seminars/intro-to-linear-regression-r/) (https://stats.oarc.ucla.edu/wp-content/uploads/2024/02/Intro-to-regression.html)
* [Linear Regression Hands on R tutorial by Princeton University Library](https://libguides.princeton.edu/R-linear_regression) (https://libguides.princeton.edu/R-linear\_regression)
* [How to Do Linear Regression in R by Datacamp](https://www.datacamp.com/tutorial/linear-regression-R) (https://www.datacamp.com/tutorial/linear-regression-R)
* The dataset used in some examples of this chapter are example data regarding the association of medical costs and patient information: [may be downloaded](https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/insurance.csv) [from Github](https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/insurance.csv) (https://raw.githubusercontent.com/stedy/Machine-Learning-with-R-datasets/master/insurance.csv).

## Chapter Summary

* *lm()* function fits linear regression which evaluates the relationship between a continuous dependent variable and one (for simple linear regression) or more (for multiple linear regression) independent variables
* *summary()* function summarizes the model results
* *coef()* to extract the model coefficients;
* *confint()* to get confidence intervals;
* *predict()* to generate predictions from the model;
* *residuals()* to access the model’s residuals
* *plot()* function applied to an lm object generates diagnostic plots
* *influence.measures()* function to detect influential data points.
* Main assumptions of linear regression (Linear relationship, Independence, Homoscedasticity, normal distribution of the outcome) should be evaluated for linear model
* Model selection may be based on AIC, BIC and may be done with *step()* function
* Model performance metrics may be estimated using “*yardstick”* package.
* Model calibration may be evaluated using calibration plot (e.g., with “*ggplot2”*)

# Logistic Regression with R

This chapter introduces the readers to performing Logistic Regression in R. It starts with addressing the fundamental challenges of using a linear model for probability and showing how the Logistic function may be used for a probability constrained between 0 and 1. We frame Logistic Regression as a specific type of Generalized Linear Models (GLM), which is fitted in R using the *glm()* function with the binomial family. The practical examples focus on building, interpreting, and evaluating Univariable and Multivariable Logistic Regression models, model evaluation using key classification metrics and discrimination power using the “*ROCR”* package as well as model selection and model validation.

## Why logistic regression is needed?

For logistic regression, the outcome (dependent variable) is categorical and logistic regression models the probability of an outcome occurring.

If Linear Regression serves to predict continuous variables, logistic Regression is used to predict categorical variables or to model the probability of an outcome occurring (binary classification as “yes” or “no”).

Now, load the required packages for this chapter (*"knitr" [5], "rmarkdown" [6], "tidyverse" [11], "arsenal” [19], “kableExtra" [12], "pscl” [39], “ROCR" [40]*):

*#load multiple packages*

Packages <- c("knitr", "rmarkdown", "tidyverse", "arsenal",

"kableExtra", "pscl", "ROCR")

lapply(Packages, library, character.only = TRUE)

Example:

The diabetes data is used to predict the occurrence of the outcome diabetes using other variables such as BMI, skin thickness, blood glucose,... This example dataset may be downloaded from github.

First, we take a look at the dataset and then examine the relationship between “bmi” and “outcome” by linear regression.

url\_base<-"https://raw.githubusercontent.com/plotly/"

data\_path<-"datasets/master/diabetes.csv"

*# Load example data*

data<-read.csv(paste0(url\_base, data\_path))

colnames(data)<-tolower(colnames(data))

str(data)

## 'data.frame': 768 obs. of 9 variables:

## $ pregnancies : int 6 1 8 1 0 5 3 10 2 8 ...

## $ glucose : int 148 85 183 89 137 116 78 115 197

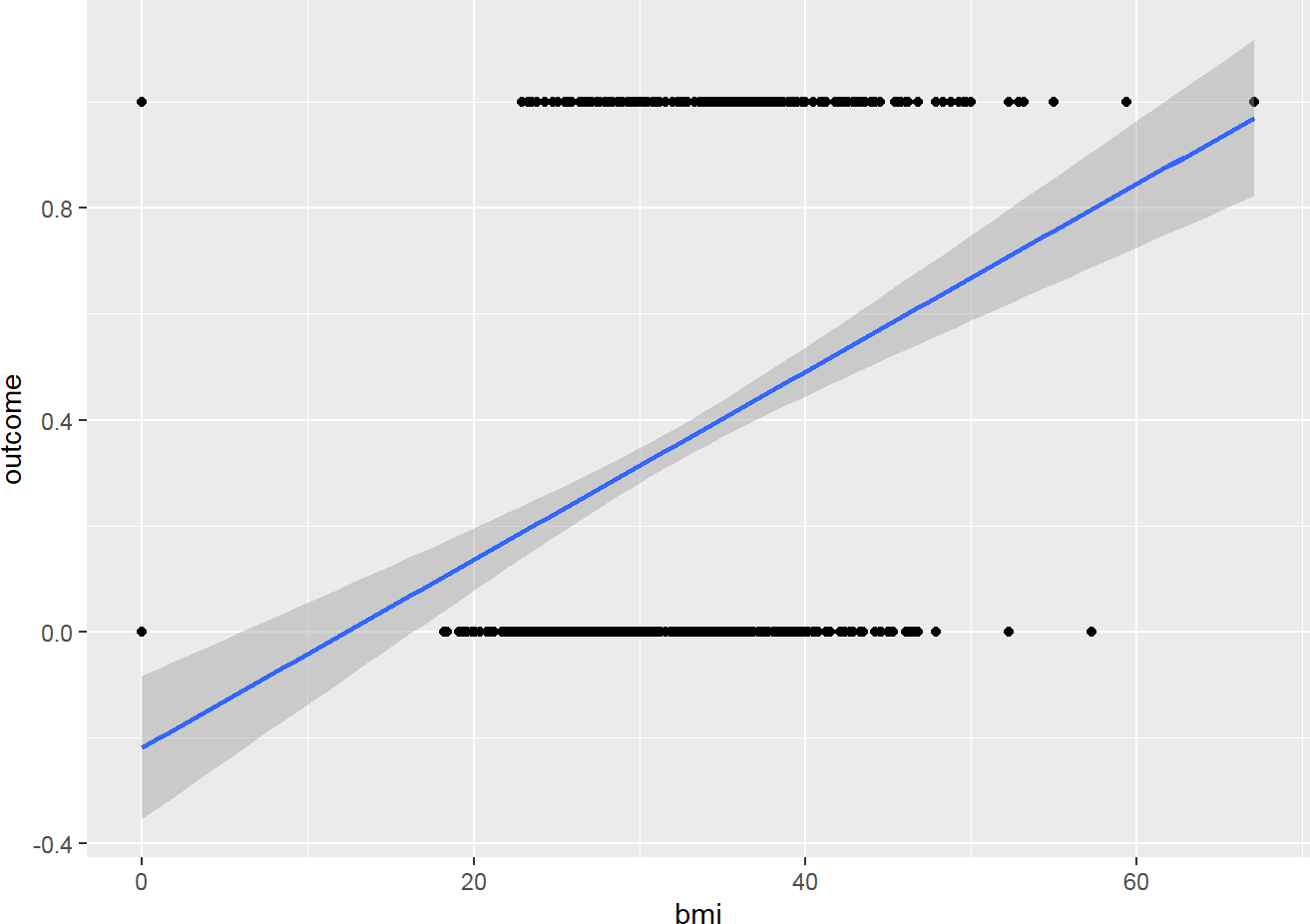
125 ...

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | $ | bloodpressure : | int | 72 66 64 66 40 74 50 0 70 96 ... | | | |
| ## | $ | skinthickness : | int | 35 29 0 23 35 0 32 0 45 0 ... | | | |
| ## | $ | insulin : | int | 0 0 0 94 168 0 88 0 543 0 ... | | | |
| ## | $ | bmi : | num | 33.6 26.6 23.3 28.1 43.1 25.6 | | | |
|  |  | 31 35.3 30.5 | | | 0 ... |  |  |
| ## | $ | diabetespedigreefunction: num 0.627 | | | 0.351 | 0.672 | 0.167 2.288 ... |
| ## | $ | age : int 50 31 | | | 32 21 | 33 30 | 26 29 53 54 ... |
| ## | $ | outcome : int 1 0 1 | | | 0 1 0 | 1 0 1 | 1 ... |

ggplot(data, aes(bmi, outcome))+

geom\_point()+

geom\_smooth(method = 'lm', formula = y~x)



Source: Figure by author(s).

**Figure 52**. Model outcome vs BMI by linear regression.

Below are explanations of what we do above:

* 0 means there is no “outcome” (diabetes), 1 means there is the “outcome” (diabetes)
* What we try to predict is the likelihood of having the “outcome” (diabetes)
* This may be represented as following:
  + y-axis being the Probability of having the “outcome” (diabetes)
    - 0 = 0% of having the “outcome” (diabetes)
    - 1 = 100% of having the “outcome” (diabetes)
  + x-axis = the “bmi” values

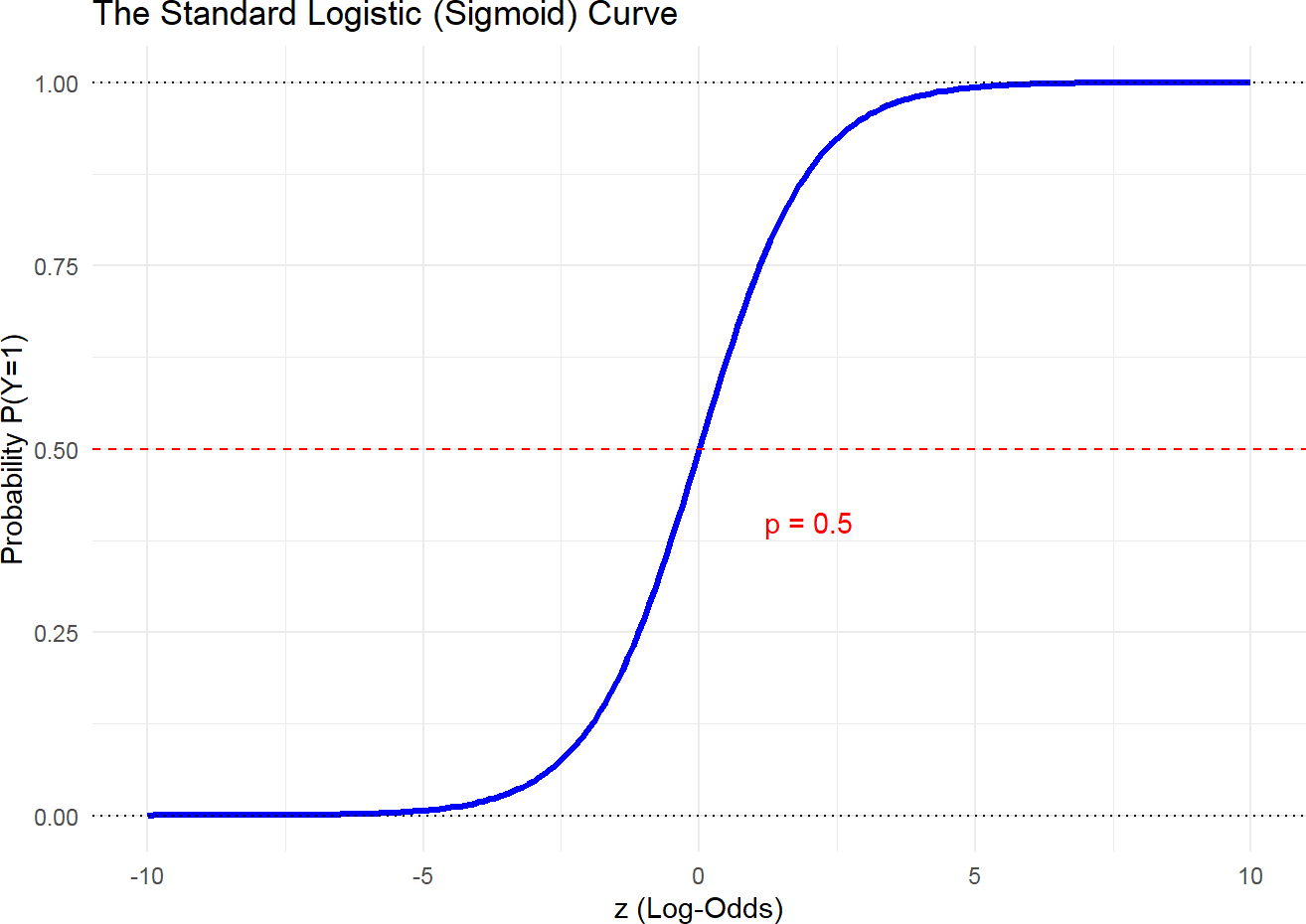
The figure above shows a bad fit line, because it cannot capture the pattern of data. It does not go through most of data points. Furthermore:

* When “bmi” decreases, we predict negative probability!
* With higher “bmi”, we even predict probability larger than 1!
* The predictions are not sensible as the probability must fall between 0 and 1 Logistic regression provides the solution for this issue.

## Logistic Regression Overview

### Statistical Concepts

The solution for the above issue is to transform a Linear Regression line to a Logistic Regression curve, which can only go between 0 and 1.



Source: Figure by author(s).

**Figure 53**. Logistic Regression curve.

Logistic function is one type of Sigmoid function [41]:

* Mathematical form:
* Graphical form: S curve

Of note, it doesn’t matter what value of Z we put into the logistic function, you will always get a value of 0 and 1, which just meets the requirements that we need. Therefore, logistic Regression function results in a probability from 0 to 1.

Logistic Regression is a specific type of Generalized Linear Models (GLM). GLM are a generalization of the concepts and abilities of regular Linear Models. Therefore, if ones are familiar with Linear Models, then ones may understand Logistic Regression.

A Logistic Regression will model the chance of an outcome based on individual characteristics. Because chance is a ratio, what will be actually modeled is the logarithm of the chance given by:

*,* where:

* + - * *y* indicates the probability of an event
      * *bi* are the regression coefficients associated with the reference group and the *xi*

explanatory variables.

* + - * The reference group, represented by *bo*, is constituted by those individuals presenting the reference level of each and every variable *x*1...*n*.

#### Plot Predicted Probabilities From a Fitted Logistic Model

Now, we go back to our previous example, using logistic regression, we plot predicted probabilities of having outcome as a function of BMI.

*# Univariable model*

logistic\_uni <- glm(outcome ~ bmi,

data = data,

family = binomial(link = "logit"))

*# 1. Create a new data frame for prediction*

*# We will vary 'glucose' across its range and # hold other predictors at their mean* new\_data <- data.frame(

bmi = data$bmi,

outcome = data$outcome)

*# 2. Use the model to predict probabilities for this new data # The 'type = "response"' argument ensures*

*# we get probabilities (0-1)*

predicted\_probs <- predict(logistic\_uni,

newdata = new\_data, type = "response")

*# 3. Add these predictions to our new data frame*

new\_data$predicted\_prob<-predicted\_probs

*# 4. Generate the plot*

ggplot(data = new\_data, aes(x = bmi, y = outcome)) +

*# Add the raw data points (0s and 1s). We use jitter to avoid overplotting.*

geom\_jitter(height = 0.05, width = 0,

alpha = 0.2, color = "gray50") +

*# Add the logistic curve from our model's predictions*

geom\_line(data = new\_data, aes(x = bmi, y = predicted\_prob), color = "blue", linewidth = 1.2)+

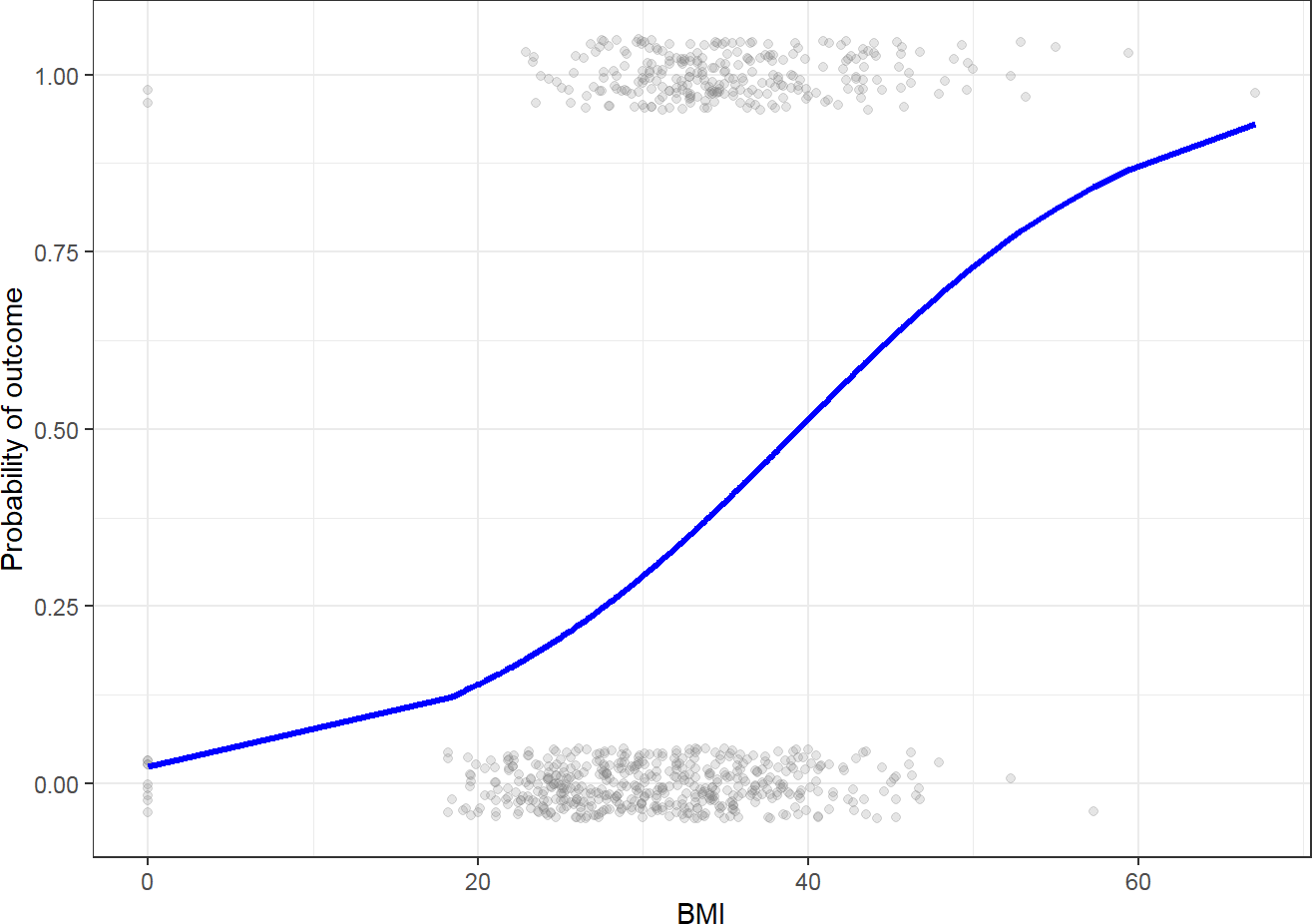
*# Add labels and a title*

labs(x = "BMI",

y = "Probability of outcome") +

*# Use a clean theme*

theme\_bw()



Source: Figure by author(s).

**Figure 54**. Predicted Probability of outcome vs. BMI.

#### Coefficients

In terms of the coefficients, Logistic Regression is similar as Linear models except the coefficients are in terms of the log(odds).

Example: Given a person profile of BMI, blood glucose, blood pressure, our aim is to build a model that predict the probability of that person getting outcome (diabetes).

This is part of the coefficients from multivariable logistic regression model fitted with *glm()* [38]:

*# Load example data*

*#data<-read.csv("https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv") #colnames(data)<-tolower(colnames(data))*

data$outcome<-as.factor(data$outcome)

*# Multivariate model*

logistic\_multi <- glm(outcome ~ bmi + glucose +

bloodpressure,data = data,

family = binomial(link = "logit")) summary(logistic\_multi )

##

## Call:

## glm(formula = outcome ~ bmi + glucose + bloodpressure, family = binomial(link = "logit"),

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ##  ## ## ## | data = data)  Coefficients:  Estimate | Std. Error | z value |  | Pr(>|z|) |
| ## | (Intercept) -7.182306 | 0.635462 | -11.303 | < | 0.0000000000000002 |
| ## | bmi 0.079793 | 0.013565 | 5.882 |  | 0.00000000405 |
| ## | glucose 0.035746 | 0.003328 | 10.740 | < | 0.0000000000000002 |
| ## | bloodpressure -0.007420 | 0.004862 | -1.526 |  | 0.127 |
| ## |  |  | | | |
| ## | (Intercept) \*\*\* |
| ## | bmi \*\*\* |
| ## | glucose \*\*\* |
| ## | bloodpressure |
| ## | --- |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for binomial family taken to be 1)

##

## Null deviance: 993.48 on 767 degrees of freedom

## Residual deviance: 769.07 on 764 degrees of freedom

## AIC: 777.07

##

## Number of Fisher Scoring iterations: 4

* + - * The Intercept gives the mean level of the response variable when all other variables are 0. The estimated intercept is −7.1823064, the standard error of the estimated intercept is 0.6354617. The z value is the estimated intercept divided by the standard error. In other words, it’s the number of standard deviations the estimated intercept is away from 0 on a standard normal curve.
      * Each one-unit change in “bmi” will increase the log(odds) of getting the “outcome”

(diabetes) by 0.0797926, and its p value indicates that it is significant in determining the “outcome”.

#### Maximum Likelihood

For Logistic Regression, maximum likelihood is used to select the correct sigmoid function that best fits the data. In other words, maximum likelihood is used to evaluate which sigmoid function would generate the observed data with the highest probability.

The example dataset above will be used again. The univariable and multivariable logistic models above will be compared regarding their maximum likelihood to evaluate which model better fit the data.

*# Extract the maximum log-likelihood for each model*

ll\_model\_1 <- logLik(logistic\_uni)

ll\_model\_2 <- logLik(logistic\_multi)

*# Create a summary data frame for easy comparison*

model\_comparison <- data.frame(

Model = c("1: Univariable Model", "2: Multivariable Model"),

LogLikelihood = c(ll\_model\_1, ll\_model\_2)

)

*# Print the comparison table, sorted by the best fit*

print(model\_comparison[order(model\_comparison$LogLikelihood,

decreasing = TRUE), ])

## Model LogLikelihood

## 2 2: Multivariable Model -384.5330

## 1 1: Univariable Model -460.3571

The result output shows that the multivariable model has larger maximum likelihood and thus better fit the data. Now, we can visualize outcome probability sigmoid curves of univariable vs. multivariable models as below:

*# Create a data frame of the original data for plotting*

plot\_df <- data.frame(outcome = as.numeric(as.character(data$outcome)),

bmi = data$bmi,

glucose = data$glucose, bloodpressure = data$bloodpressure)

*# Predicted outcome probability of univariable with only BMI*

plot\_df$predicted\_probs\_uni <- predict(logistic\_uni,

newdata = data.frame(bmi = data$bmi), type = "response")

*# Predicted outcome probability of multivariable with*

*BMI holding glucose and bloodpressure constant at their means*

plot\_df$predicted\_probs\_multi <- predict(logistic\_multi,

newdata = data.frame( bmi =

data$bmi,

glucose = mean(data$glucose),

bloodpressure = mean(data$bloodpressure)),

type = "response")

ggplot(plot\_df, aes(x = bmi, y = outcome)) +

*# Add the raw data points (0s and 1s) with jitter* geom\_jitter(height = 0.05, width = 0, alpha = 0.2) + *# --- Add the two probability curves ---*

*# Curve for univariable Model 1 (BMI Only)*

geom\_line(aes(x = bmi, y = predicted\_probs\_uni, color = "Model 1: Univariable"))+

*# Curve for multivariable Model 2*

*# (with BMI, holding glucose and bloodpressure at their means)*

geom\_line(aes(x = bmi, y = predicted\_probs\_multi, color = "Model 2: Multivariable"))+

*# --- Formatting and Labels ---*

labs(

x = "BMI",

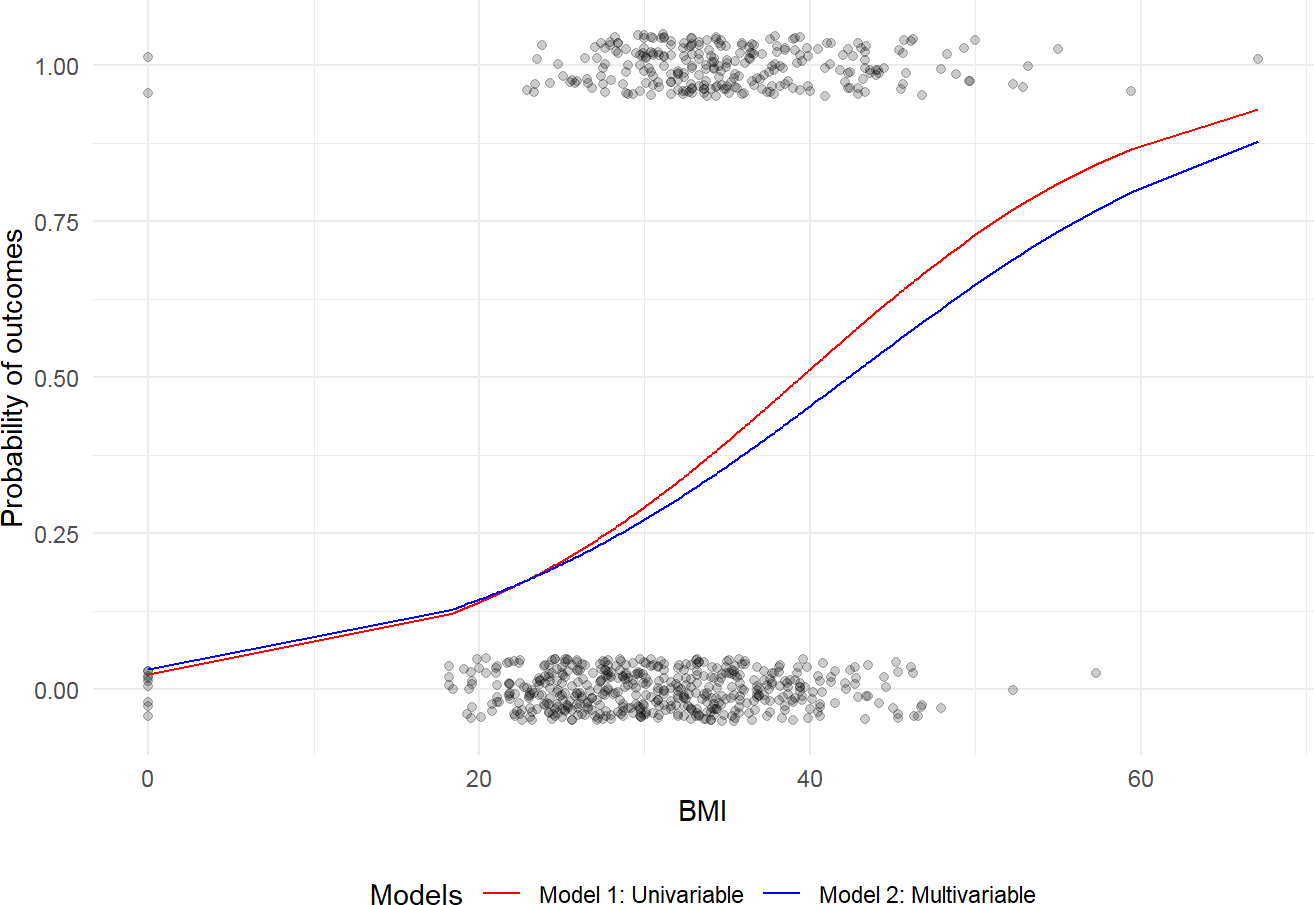
y = "Probability of outcomes", color = "Models" *# Legend title*

) +

scale\_color\_manual(values = c( "Model 1: Univariable" = "red", "Model 2: Multivariable" = "blue"

)) +

theme\_minimal() + theme(legend.position = "bottom")



Source: Figure by author(s).

**Figure 55**. Sigmoid curves of univariable vs. multivariable models.

#### R-Squared

The R-squared from linear regression (the proportion of variance explained) cannot be directly applied for logistic regression because logistic regression does not work by minimizing squared errors. Instead, it uses Maximum Likelihood Estimation (MLE). To fill this gap, several pseudo R-squared measures have been developed. One of the most commonly used is McFadden’s R-squared. McFadden’s R squared measure is defined as:

where Lc denotes the (maximized) likelihood value from the current fitted model, and Lnull denotes the corresponding value for the null model (the model with only an intercept and no covariates).

The log-likelihood *R*2 values go from 0, for poor models, to 1, for good models.

As a rule of thumb, values between 0.2 and 0.4 may be considered to represent a good model fit.

Below is an example to calculate McFadden’s *R*2 manually:

*# Method 1: Manual Calculation (for understanding)*

*# 1. Fit the "null model" (intercept only)*

null\_model <- glm(outcome ~ 1,

data = data,

family = binomial(link = "logit"))

*# 2. Extract the log-likelihood from both models*

logLik\_full <- logLik(logistic\_multi)

logLik\_null <- logLik(null\_model)

*# 3. Apply McFadden's formula*

mcfadden\_r2 <- 1 - (logLik\_full / logLik\_null)

cat("McFadden's R-squared (manual calculation):",

as.numeric(mcfadden\_r2), "\n")

## McFadden's R-squared (manual calculation): 0.2258898

Below is an example to calculate McFadden’s *R*2 using the *pR2()* function of the “*pscl”* package (The *pR2()* function calculates several pseudo R-squared values) [39]:

pseudo\_r2\_values <- pR2(logistic\_multi) ## fitting null model for pseudo-r2

print(pseudo\_r2\_values)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | llh | llhNull | G2 | McFadden | r2ML |
| ## | -384.5330385 -496.7419551 224.4178332 | | | 0.2258898 | 0.2533888 |
| ## | r2CU | | |  |  |
| ## | 0.3491560 | | |  |  |

The output shows that the multivariable model has a relatively good model fit and has a 0.2258898 better fit than the null model.

### Univariable and Multivariable Models

In Logistic Regression the outcome or dependent variable is binary. The predictor or independent variable is one with univariable model and more than one with multivariable model. In reality, most outcomes have many predictors. Hence, multivariable Logistic Regression is closer to reality.

## Logistic Regression Model Building and Selection

### Main Steps

Below are the main steps of building Logistic Regression (logit) models:

* + - * Import data
      * Clean data and check for class bias
      * Create training and test samples
      * Compute information value to find out important variables
      * Build logit models
      * Predict on test data
      * Do model evaluation

### Building Model in R

We use the same example diabetes dataset above. Our aim is to build a model so that we can predict the probability of having the “outcome” (diabetes) if a profile is given.

#### Import Data

Now we import the dataset and take a look at the data’s structure by using function *str()* and summarize all variables in the dataset by “outcome” and overall:

url\_base<-"https://raw.githubusercontent.com/plotly/"

data\_path<-"datasets/master/diabetes.csv"

*# Load example data*

data<-read.csv(paste0(url\_base, data\_path))

colnames(data)<-tolower(colnames(data))

str(data)

## ‘data.frame’: 768 obs. of 9 variables: $ pregnancies : int 6 1 8 1 0 5 3 10 2 8 . . .

## $glucose : int 148 85 183 89 137 116 78 115 197 125 . . .

## $ bloodpressure : int 72 66

## 64 6640 74 50 0 70 96 . . .

## $ skinthickness : int 35 29 0 23 35 0 32 0 45 0 . . .

## $ insulin : int 0 0 0 94 168 0 88 0 543 0 . . .

## $ bmi : num 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 . . .

## $ diabetespedigreefunction: num 0.627 0.351 0.672 0.167 2.288 . . .

## $ age : int 50 31 32 21 33 30 26 29 53 54 . . .

## $ outcome : int 1 0 1 0 1 0 1 0 1 1 . . .

*#convert outcome to factor*

data$outcome<-as.factor(data$outcome)

*#summary of all variables in the dataset using the arsenal package*

my\_controls <- tableby.control( test = T,

total = T,

numeric.stats = c("meansd", "medianq1q3","meanCI","range","Nmiss2"), cat.stats = c("countpct", "Nmiss2"),

stats.labels = list( meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)", range = "Min - Max",

Nmiss2 = "Missing", meanCI = "Mean (95%CI)"

)

)

prevar<-colnames(data)[!colnames(data) %in% "outcome"] mylabels <-as.list(prevar)

names(mylabels)<-prevar

tab <- tableby(as.formula(paste("outcome",

paste(prevar,collapse = "+"),sep = "~")),

data = data,

control = my\_controls) kable(summary(tab,labelTranslations = mylabels, text = TRUE),

caption = "Data summary")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 8**. Data summary.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0 (N = 500)** | **1 (N = 268)** | **Total (N = 768)** | **p Value** |
| pregnancies |  |  |  | < 0.001 |
| - Mean (SD) | 3.298 (3.017) | 4.866 (3.741) | 3.845 (3.370) |  |
| - Median (Q1, Q3) | 2.000 (1.000, 5.000) | 4.000 (1.750, 8.000) | 3.000 (1.000, 6.000) |  |
| - Mean (95%CI) | 3.298 (3.033, 3.563) | 4.866 (4.416, 5.316) | 3.845 (3.606, 4.084) |  |
| - Min−Max | 0.000−13.000 | 0.000−17.000 | 0.000−17.000 |  |
| - Missing | 0 | 0 | 0 |  |
| glucose |  |  |  | < 0.001 |
| - Mean (SD) | 109.980 (26.141) | 141.257 (31.940) | 120.895 (31.973) |  |
| - Median (Q1, Q3) | 107.000 (93.000, 125.000) | 140.000 (119.000, 167.000) | 117.000 (99.000, 140.250) |  |
| - Mean (95%CI) | 109.980 (107.683, 112.277) | 141.257 (137.416, 145.099) | 120.895 (118.630, 123.159) |  |
| - Min−Max | 0.000−197.000 | 0.000−199.000 | 0.000−199.000 |  |
| - Missing | 0 | 0 | 0 |  |
| bloodpressure |  |  |  | 0.072 |
| - Mean (SD) | 68.184 (18.063) | 70.825 (21.492) | 69.105 (19.356) |  |
| - Median (Q1, Q3) | 70.000 (62.000, 78.000) | 74.000 (66.000, 82.000) | 72.000 (62.000, 80.000) |  |
| - Mean (95%CI) | 68.184 (66.597, 69.771) | 70.825 (68.240, 73.409) | 69.105 (67.734, 70.477) |  |
| - Min−Max | 0.000−122.000 | 0.000−114.000 | 0.000−122.000 |  |
| - Missing | 0 | 0 | 0 |  |
| skinthickness |  |  |  | 0.038 |
| - Mean (SD) | 19.664 (14.890) | 22.164 (17.680) | 20.536 (15.952) |  |
| - Median (Q1, Q3) | 21.000 (0.000, 31.000) | 27.000 (0.000, 36.000) | 23.000 (0.000, 32.000) |  |
| - Mean (95%CI) | 19.664 (18.356, 20.972) | 22.164 (20.038, 24.291) | 20.536 (19.406, 21.666) |  |
| - Min−Max | 0.000−60.000 | 0.000−99.000 | 0.000−99.000 |  |
| - Missing | 0 | 0 | 0 |  |

**Table 8.** *Cont.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **0 (N = 500)** | **1 (N = 268)** | **Total (N = 768)** | **p Value** |
| insulin |  |  |  | < 0.001 |
| - Mean (SD) | 68.792 (98.865) | 100.336 (138.689) | 79.799 (115.244) |  |
| - Median (Q1, Q3) | 39.000 (0.000, 105.000) | 0.000 (0.000, 167.250) | 30.500 (0.000, 127.250) |  |
| - Mean (95%CI) | 68.792 (60.105, 77.479) | 100.336 (83.656, 117.016) | 79.799 (71.636, 87.963) |  |
| - Min−Max | 0.000−744.000 | 0.000−846.000 | 0.000−846.000 |  |
| - Missing | 0 | 0 | 0 |  |
| bmi |  |  |  | < 0.001 |
| - Mean (SD) | 30.304 (7.690) | 35.143 (7.263) | 31.993 (7.884) |  |
| - Median (Q1, Q3) | 30.050 (25.400, 35.300) | 34.250 (30.800, 38.775) | 32.000 (27.300, 36.600) |  |
| - Mean (95%CI) | 30.304 (29.629, 30.980) | 35.143 (34.269, 36.016) | 31.993 (31.434, 32.551) |  |
| - Min−Max | 0.000−57.300 | 0.000−67.100 | 0.000−67.100 |  |
| - Missing | 0 | 0 | 0 |  |
| diabetespedigreefunction |  |  |  | < 0.001 |
| - Mean (SD) | 0.430 (0.299) | 0.550 (0.372) | 0.472 (0.331) |  |
| - Median (Q1, Q3) | 0.336 (0.230, 0.562) | 0.449 (0.263, 0.728) | 0.372 (0.244, 0.626) |  |
| - Mean (95%CI) | 0.430 (0.403, 0.456) | 0.550 (0.506, 0.595) | 0.472 (0.448, 0.495) |  |
| - Min−Max | 0.078−2.329 | 0.088−2.420 | 0.078−2.420 |  |
| - Missing | 0 | 0 | 0 |  |
| age |  |  |  | < 0.001 |
| - Mean (SD) | 31.190 (11.668) | 37.067 (10.968) | 33.241 (11.760) |  |
| - Median (Q1, Q3) | 27.000 (23.000, 37.000) | 36.000 (28.000, 44.000) | 29.000 (24.000, 41.000) |  |
| - Mean (95%CI) | 31.190 (30.165, 32.215) | 37.067 (35.748, 38.386) | 33.241 (32.408, 34.074) |  |
| - Min−Max | 21.000−81.000 | 21.000−70.000 | 21.000−81.000 |  |
| - Missing | 0 | 0 | 0 |  |

Source: Table by author(s).

We can notice that there are a greater number of “outcome” = 0 (no diabetes) than “outcome” = 1 (diabetes). From the summary table, we can see the variables showing unbalance between “outcome” = 0 (no diabetes) and “outcome” = 1 (diabetes).

#### Build Logistic Regression Model in R

For simplicity, we will build our model with all the data (no train-test splitting). For predictive modeling, we will need to do develop model on training data and validate model on test data.

The *glm()* procedure with *family = "binomial"* will build the Logistic Regression model on the given formula.

*# Univariate model*

fitu <- glm(outcome ~ bmi,data = data, family = "binomial")

*# Multivariate model*

fitm <- glm(outcome ~ bmi + glucose + bloodpressure, data = data, family = "binomial")

* *data = data* is data frame named data
* *family = "binomial"* is for Logistic Regression model
* *outcome* is the column which contains the response variable “outcome”
* ~ help us select the predictors:
  + . means chose all the data that we have.
  + If we want to chose specific variables as predictors, we can replace . with

the combination of our interested variables. For example, *~ bmi* for chosing only “bmi”; *~ bmi + glucose + bloodpressure* for chosing “bmi”, “glucose”, “bloodpressure”.

* + You can even exclude the predictor. For example, you write *~. - glucose*

to unselect variable “glucose” and end up with *fitm <- glm(outcome ~.*

*- glucose, data = data, family = "binomial")*

#### Interpretation of Results

summary(fitm) ##

## Call:

## glm(formula = outcome ~ bmi + glucose + bloodpressure, family = "binomial",

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ##  ## ## ## | data = data)  Coefficients:  Estimate | Std. Error | z value |  | Pr(>|z|) |
| ## | (Intercept) -7.182306 | 0.635462 | -11.303 | < | 0.0000000000000002 |
| ## | bmi 0.079793 | 0.013565 | 5.882 |  | 0.00000000405 |
| ## | glucose 0.035746 | 0.003328 | 10.740 | < | 0.0000000000000002 |
| ## | bloodpressure -0.007420 | 0.004862 | -1.526 |  | 0.127 |
| ## |  |  | | | |
| ## | (Intercept) \*\*\* |
| ## | bmi \*\*\* |
| ## | glucose \*\*\* |
| ## | bloodpressure |
| ## | --- |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for binomial family taken to be 1)

##

## Null deviance: 993.48 on 767 degrees of freedom

## Residual deviance: 769.07 on 764 degrees of freedom

## AIC: 777.07

##

## Number of Fisher Scoring iterations: 4

* Deviance Residuals: Summary of the distribution of the deviance residuals. Deviance residuals measure how well the observations fit the model. The closer a residual to 0 the better the fit of the observation.
* Coefficients:
  + In general, the Intercept gives the mean level of the response variable when all other variables are 0. The estimated intercept is −7.1823064, the standard error of the estimated intercept is 0.6354617, and the z value is the estimated intercept divided by the standard error.
  + Each one-unit change in “bmi” will increase the log(odds) of getting “outcome” by 0.0797926, and its p value indicates that it is significant in determining the “outcome”.
  + Each unit increase in “glucose” increases the log odds of getting “outcome” by 0.0357463 and p value indicates that it is significant in determining the “outcome”.
  + “bloodpressure” is not significant in determining the outcome.
* Deviance: Null deviance is the value when there is only intercept and no variables. Residual deviance is the value when taking all the variables into account. It makes sense to consider the model good if that difference is big enough. Generally, the degrees of freedom reported on the Null deviance are always higher than the degrees of freedom reported on the Residual deviance. The difference between Null deviance and Residual deviance tells how good the model fit is: Greater the difference better the model.
* Akaike Information Criterion (AIC): The Akaike Information Criterion (AIC) provides a method for assessing the quality of the model through comparison of related models. It’s useful for comparing models, but isn’t interpretable on its own. If there are several candidate models, the model that has the smallest AIC should be selected.

#### Making Prediction

Let’s predict if a person has a profile of “bmi” = 29, “glucose” = 148, “bloodpressure” = 96, what is the probability of that person getting “outcome” = 1.

x <- data.frame(bmi = 29, glucose = 148, bloodpressure = 96)

p<- predict(fitm,x, type = "response")

p

## 1

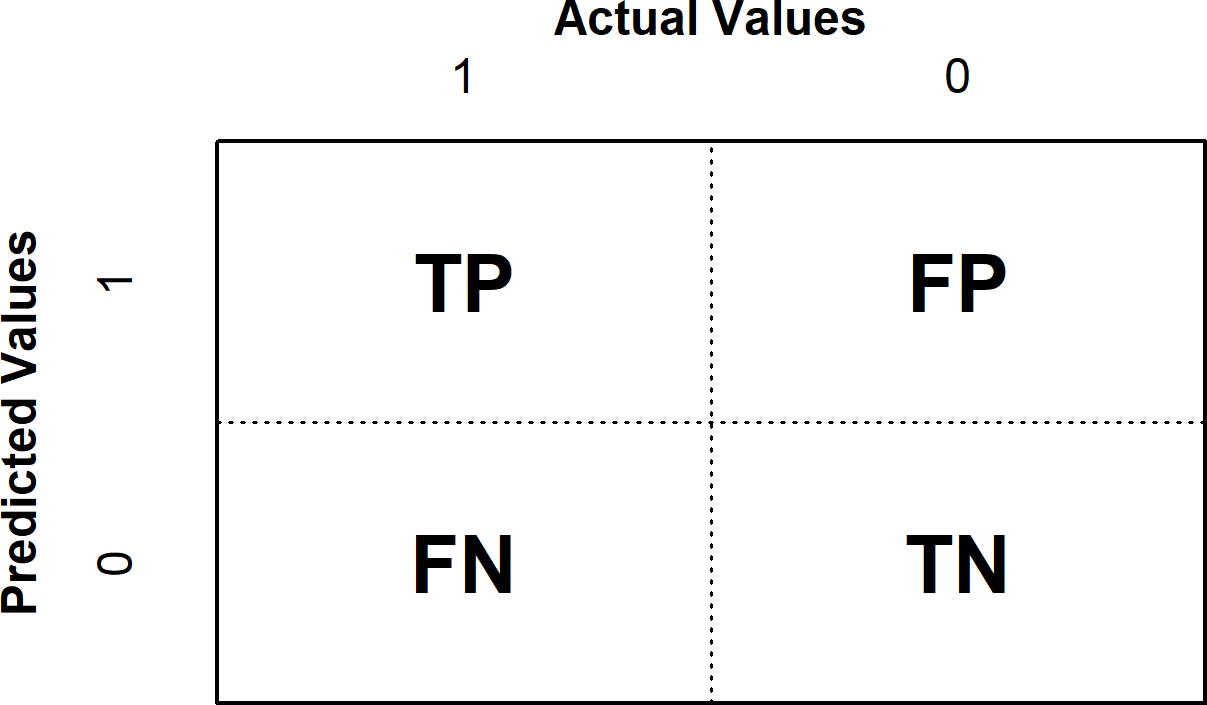
## 0.4279538

We see that there is 0.4279538 chance that she will get “outcome” = 1.

## Model Evaluation

### Classification Metrics

Classification results may be visualized using confusion matrix:



Source: Figure by author(s).

**Figure 56**. Confusion matrix.

The Positive/Negative label refers to the predicted outcome of an experiment, while the True/False refers to the actual outcome.

* + - * Type 1 error corresponds to a False Positive.
      * Type 2 error corresponds to a False Negative.

Which aspect to improve depends on specific problems. For example, for a model to identify if a breast cancer is benign or malignant, one would prefer Type 2 error to Type 1 error because one would not want to miss malignant cases.

From the confusion matrix, different metrics can be measured for the validity of the model:

* + - * Accuracy(all correct/all): ACC = (TP + TN)/(TP + TN + FP + FN)
      * Misclassificationor Error rate(all incorrect/all): Misclassification = (FP + FN)/(TP + TN + FP + FN)
      * Precisionor Positive predictive value(PPV) (true positives/predicted positives): Precision = TP/(TP + FP)
      * Sensitivityor Recall or true positive rate (true positives/all actual positives): Sensitivity refers to the test’s ability to correctly detect ill patients who do have the condition. Sensitivity = TP/(TP + FN)
      * Specificity(true negatives/all actual negatives): A positive result in a test with

high specificity is useful for ruling in disease. The test rarely gives positive results in healthy patients. A positive result signifies a high probability of the presence of disease. Specificity = TN/(TN + FP)

The accuracy score might not be the best metrics to use for a classification problem, especially if you deal with the unbalanced dataset.

### Calculate, Visualize Classification Measures

We can calculate and visualize classification measures using the “*ROCR”* package [40]. First, we use *predict()* to get predicted values from logistic regression fit object, then create a prediction object with *prediction()*, then use *performance()* to get classification measures and use *plot()* to visualize the classification measures.

#### Sensitivity by cutoff

*#library(ROCR)*

predicted <- predict(fitm, type = "response")

*# Create a prediction object*

pred <- prediction(predicted, data$outcome)

*# Sensitivity by cutoff*

perf <- performance(pred,"sens","cutoff")

plot(perf)



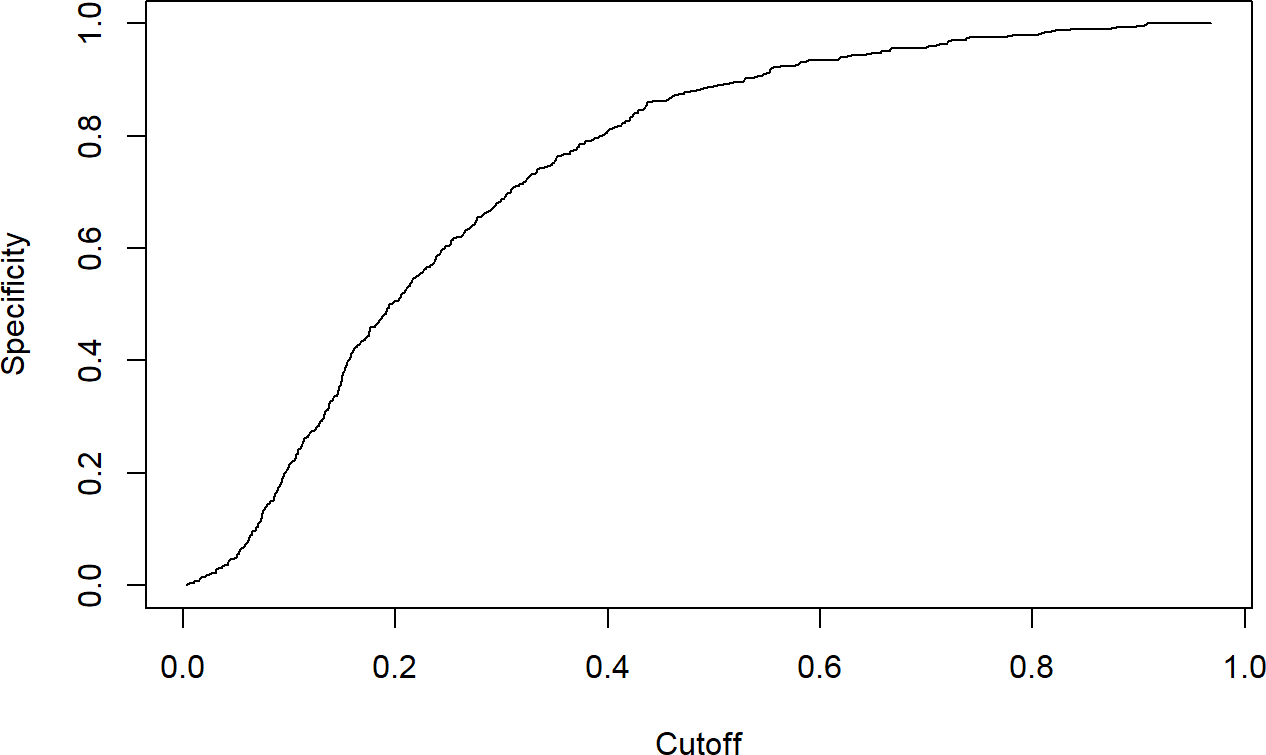
Source: Figure by author(s).

**Figure 57**. Sensitivity by cutoff.

The plot above shows that as the cutoff (x-axis) decreases (we become less strict), sensitivity (y-axis) increases and vice versa.

#### Specificity by cutoff

perf <- performance(pred,"spec","cutoff") plot(perf)



Source: Figure by author(s).

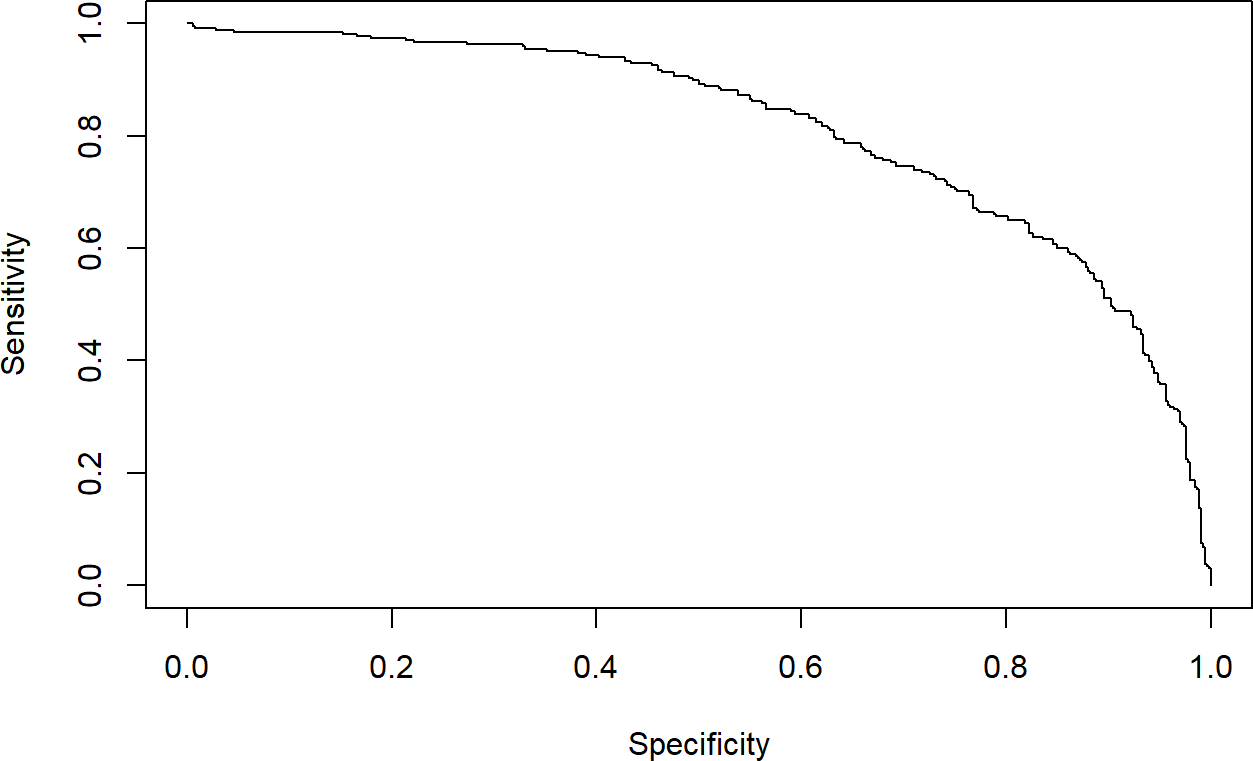
**Figure 58**. Specificity by cutoff.

The plot above shows that as the cutoff (x-axis) decreases, specificity (y-axis) decreases and vice versa.

#### Sensitivity and specificity

perf <- performance(pred, "sens", "spec")

plot(perf)



Source: Figure by author(s).

**Figure 59**. Sensitivity and specificity.

The plot above shows the change of sensitivity according specificity: as specificity increases, sensitivity decreases and vice versa.

#### The highest sensitivity + specificity

[max(perf@x.values[[](mailto:max(perf@x.values)1]][+perf@y.values[[](mailto:%2Bperf@y.values)1]]) ## [1] 1.467522

The cutoff that yields the highest sensitivity + specificity

[perf@alpha.values[[](mailto:perf@alpha.values)1]][[which.max(perf@x.values[[](mailto:which.max(perf@x.values)1]][+perf@y.values[[](mailto:%2Bperf@y.values)1]])]

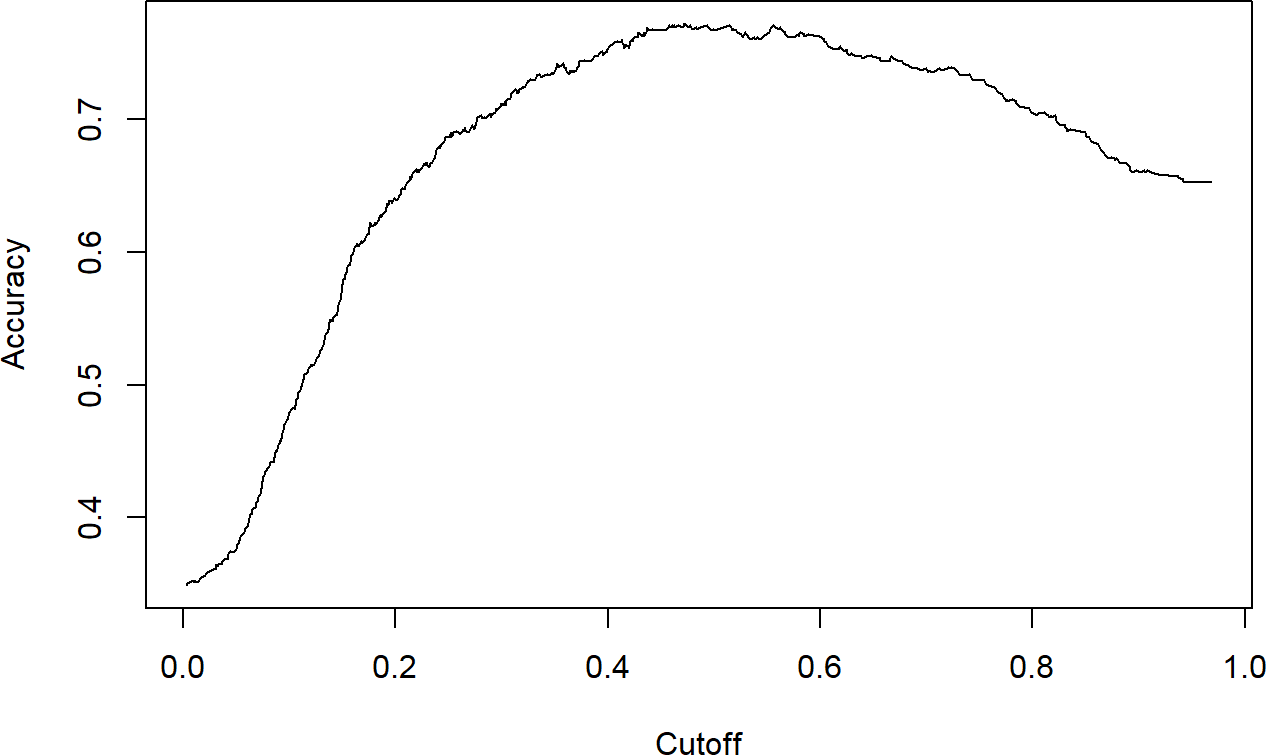
## [1] 0.4132145

The cutoff that best balances sensitivity and specificity is 0.4132145.

Accuracy (all correct/all) by cutoff

perf <- performance(pred,"acc","cutoff")

plot(perf)



Source: Figure by author(s).

**Figure 60**. Accuracy by cutoff.

The plot above shows that the model's overall accuracy is highest with a cutoff somewhere between 0.4 and 0.6.

### Discrimination Power

Discrimination power may be evaluated using Receiver Operating Characteristics (ROC) Curve and Area Under the Curve (AUC).

#### Receiver Operating Characteristics Curve (ROC)

Receiver Operating Characteristics Curve (ROC) traces the percentage of TP accurately predicted by a given model as the prediction probability cutoff is lowered from 1 to 0. It plots Sensitivity (TPR) on the y-axis against 1 - Specificity (FPR) on the x-axis for all possible cutoffs. We will mostly use the R package “ROCR” for evaluation of model performance [40].

*#library(ROCR)*

predicted <- predict(fitm, type = "response")

*# Create a prediction object*

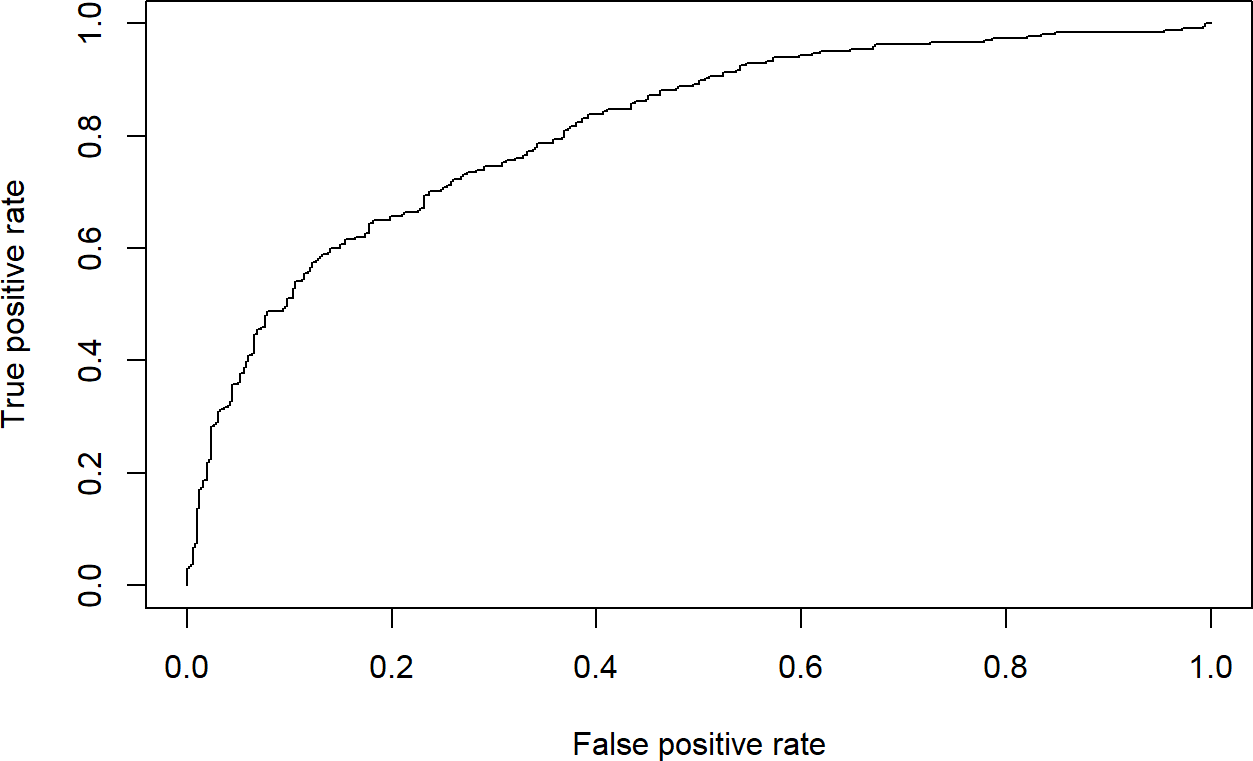
pred <- prediction(predicted, data$outcome)

*# Create a performance object*

perf <- performance(pred,"tpr","fpr")

*# ROC curve*

plot(perf)



Source: Figure by author(s).

**Figure 61**. Receiver Operating Characteristics Curve (ROC).

For a good model:

* + - * As the cutoff is lowered, it should mark more of actual label 1 as positives and lesser of actual label 0 as label 1.
      * The curve should rise steeply, indicating that the TPR (Y-Axis)(or Sensitivity) increases faster than the FPR (X-Axis)(or Specificity) as the cutoff score decreases.
      * A model with no discrimination power would be a 45-degree diagonal line.
      * A perfect model would go straight up to the top-left corner (100% sensitivity, 100% specificity) and then across.
      * Our model's curve bows far up to the top-left, showing it has good discriminatory power. It is much better than random chance.

#### Area Under the Curve (AUC)

The Area Under the Curve (AUC) is a single number that summarizes the entire ROC curve. It represents the probability that the model will rank a randomly chosen positive case higher than a randomly chosen negative case. The greater the area under the ROC curve (AUC), the better the predictive ability of the model.

• AUC = 1.0: Perfect model.

• AUC = 0.5: Useless model (same as random chance).

• AUC > 0.7: Generally considered a good model.

perf <- performance(pred,"auc") [perf@y.values[[](mailto:perf@y.values)1]]

## [1] 0.8114627

An AUC of 0.8115 indicates good discriminatory power and the model is very effective at distinguishing between patients with and without diabetes.

### Model Fit Statistics

We may evaluate logistic regression model goodness of fit using some model fit statistics below.

#### Akaike Information Criterion (AIC)

A measure of model fit that penalizes complexity. Lower values are better when comparing different models. We can get AIC using *AIC()*.

aic\_value <- AIC(logistic\_uni)

cat("AIC univariable model: ", round(aic\_value, 2), "\n")

## AIC univariable model: 924.71

aic\_value <- AIC(logistic\_multi)

cat("AIC multivariable model: ", round(aic\_value, 2), "\n")

## AIC multivariable model: 777.07

The output shows that the multivariable model has better model fit than the univariable model.

#### Bayesian Information Criterion (BIC)

Similar to AIC but with a larger penalty for model complexity. The model with lower BIC is better. We can get BIC using *BIC()*.

bic\_value <- BIC(logistic\_uni)

cat("BIC univariable model: ", round(bic\_value, 2), "\n")

## BIC univariable model: 934

bic\_value <- BIC(logistic\_multi)

cat("BIC multivariable model: ", round(bic\_value, 2), "\n")

## BIC multivariable model: 795.64

The output also shows that the multivariable model has better model fit than the univariable model.

#### McFadden’s Pseudo R-Squared

Using the *pR2()* function from the “*pscl”* package [39]:

mcfadden\_r2 <- pR2(logistic\_multi)["McFadden"]

## fitting null model for pseudo-r2

cat("McFadden's Pseudo R-squared for multivariable model: ", round(mcfadden\_r2, 4), "\n")

## McFadden's Pseudo R-squared for multivariable model: 0.2259

The output shows the proportional improvement in model fit compared to a null model. A value of 0.2259 suggests a relatively good improvement.

## Model Validation

### Create Training and Test Samples

For model validation, the idea is that we’ll like to divide our original data into two groups: the training for training our model. While we’ll leave apart a bunch of data to check if our model is consistent. This will be our test group. And if we obtain a much worse prediction on our test set than in our train group, then probably we’re overfitting our training data. Usually an 80%–20% or 70%–30% train-test split is considered reasonable.

Because there are a greater number of “outcome” = 0 (no diabetes) than “outcome” = 1 (diabetes), we can address this problem of class bias by building model from a balance *trainingData*.

*#library(tidyverse)*

data$rn<-paste("rn",1:nrow(data),sep = "") *#to facilitate row selecting later*

set.seed(100) *# for repeatability of samples # Create Training data*

*# (sampling with the same fraction 0.8 for outcome = 0 (no diabetes) # and outcome = 1 (diabetes)*

trainingData<-data %>% group\_by(outcome) %>%

sample\_frac(size = 0.8) *# sampling fraction the same*

*# Test Data is the remaining data that is not in training data*

testData <- data[!data$rn %in% trainingData$rn,]

*# drop rn*

trainingData<-trainingData %>% select(!rn)

testData<-testData %>% select(!rn)

table(trainingData$outcome)

##

## 0 1

## 400 214

table(testData$outcome) ##

## 0 1

## 100 54

*group\_by(outcome)* tells *sample\_frac*() to operate within each outcome group separately. *sample\_frac(size = 0.8)* randomly selects 80% of the rows from that group. This results in a training set with 400 "0"s and 214 "1"s, and a test set with 100 "0"s and 54 "1"s, perfectly preserving the original ratio or similar proportion of “outcome” = 0 and “outcome” = 1 between *trainingData* and *testData*.

### Build Logitic Models on Training Data and Predict on Test Data

We build the same multivariable model above that uses “bmi”, “glucose”, “bloodpressure” to predict “outcome” on *trainingData* then validate the model on *testData*. The model on *trainingData*:

fitm.train<- glm(outcome ~ bmi + glucose + bloodpressure,

data = trainingData, family = "binomial")

summary(fitm.train)

##

## Call:

## glm(formula = outcome ~ bmi + glucose + bloodpressure, family = "binomial",

## data = trainingData)

##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ##  ## | Coefficients: | Estimate | Std. Error | z value | Pr(>|z|) |
| ## | (Intercept) | -7.385483 | 0.716207 | -10.312 < | 0.0000000000000002 |
| ## | bmi | 0.081722 | 0.015328 | 5.332 | 0.0000000973 |
| ## | glucose | 0.036485 | 0.003691 | 9.886 < | 0.0000000000000002 |
| ## | bloodpressure | -0.006937 | 0.005440 | -1.275 | 0.202 |
| ## |  |  |  | | |
| ## | (Intercept) | \*\*\* |
| ## | bmi | \*\*\* |
| ## | glucose | \*\*\* |
| ## | bloodpressure |  |
| ## | --- |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' | ' | 1 |
| ## |  |  |  |
| ## | (Dispersion parameter for binomial family taken to be 1) |  |  |
| ## |  |  |  |
| ## | Null deviance: 793.94 on 613 degrees of freedom |  |  |
| ## | Residual deviance: 605.86 on 610 degrees of freedom |  |  |
| ## | AIC: 613.86 |  |  |
| ## |  |  |  |
| ## | Number of Fisher Scoring iterations: 4 |  |  |

Now, we make prediction and evaluate model performance on testData :

*#library(ROCR)*

*# predicted probability*

predicted <- plogis(predict(fitm.train, testData))

*# or*

predicted <- predict(fitm.train, testData, type = "response")

*# Create a prediction object*

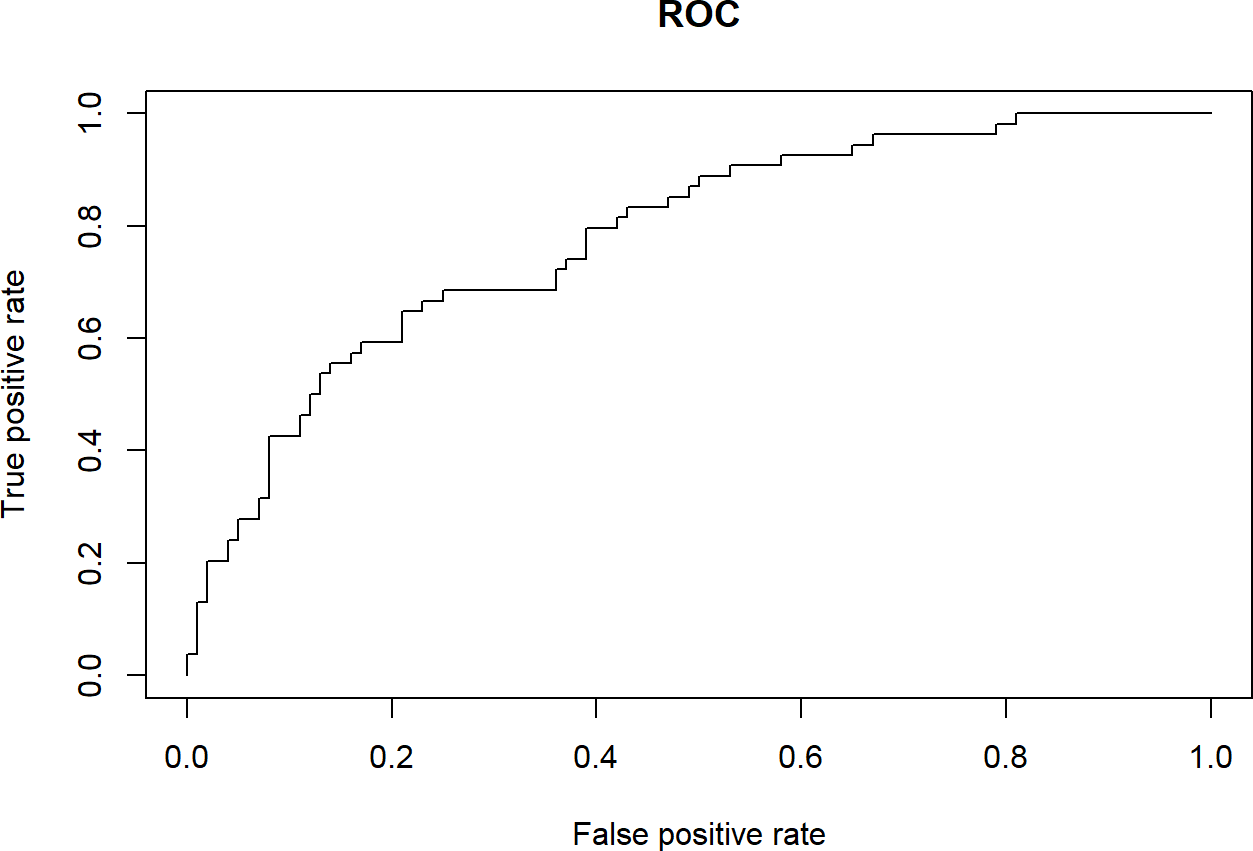
pred <- prediction(predicted, testData$outcome)

*# Create a performance object*

perf <- performance(pred,"tpr","fpr")

*# ROC curve*

plot(perf, main = "ROC")



Source: Figure by author(s).

**Figure 62**. ROC test data.

We can also check other performance measures similarly to the examples above:

*# Sensitivity by cutoff*

perf <- performance(pred,"sens","cutoff")

plot(perf)



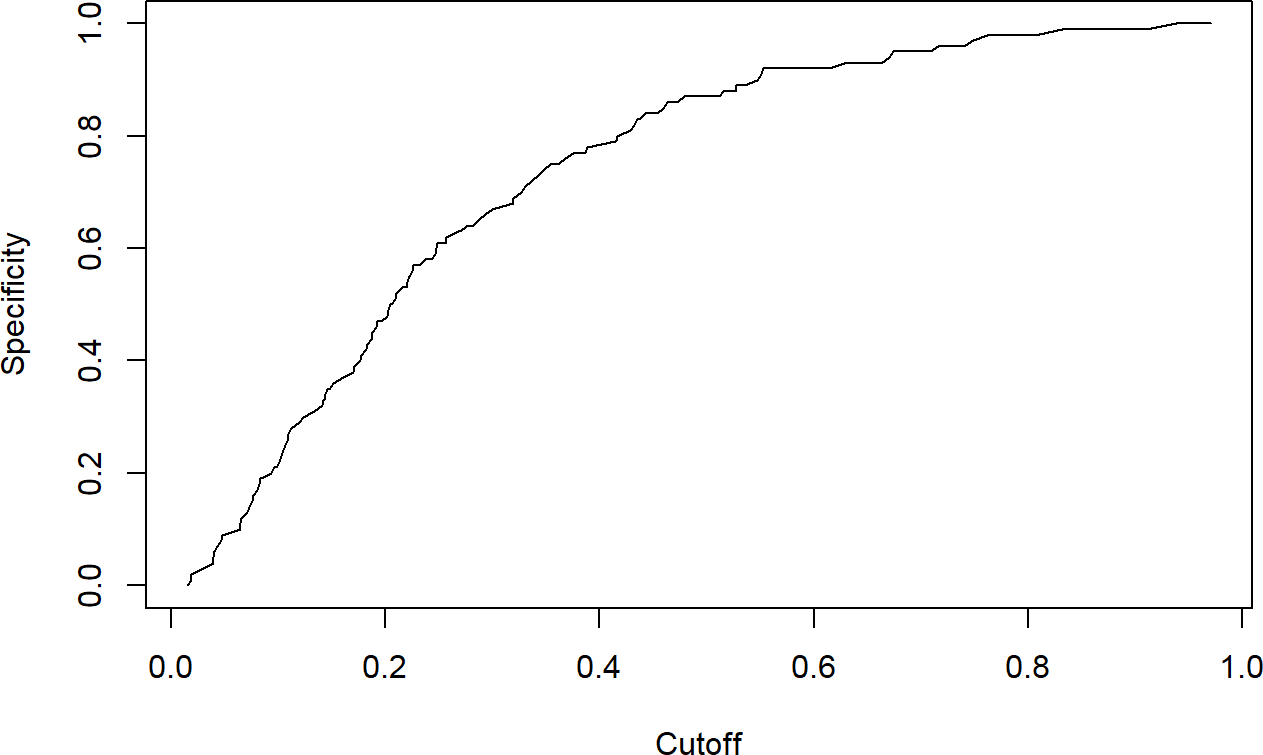
Source: Figure by author(s).

**Figure 63**. Sensitivity by cutoff test data.

*# Specificity by cutoff*

perf <- performance(pred,"spec","cutoff")

plot(perf)



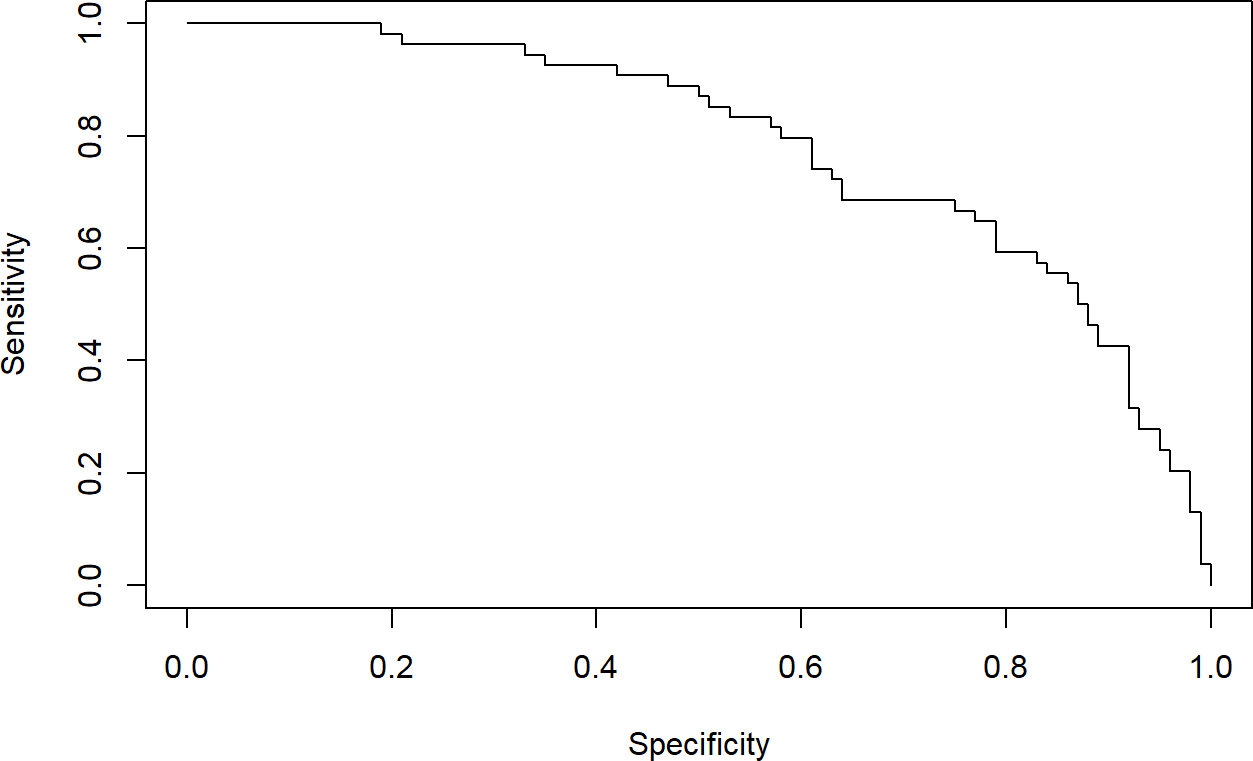
Source: Figure by author(s).

**Figure 64**. Specificity by cutoff test data.

*# Sensitivity, specificity*

perf <- performance(pred, "sens", "spec")

plot(perf)



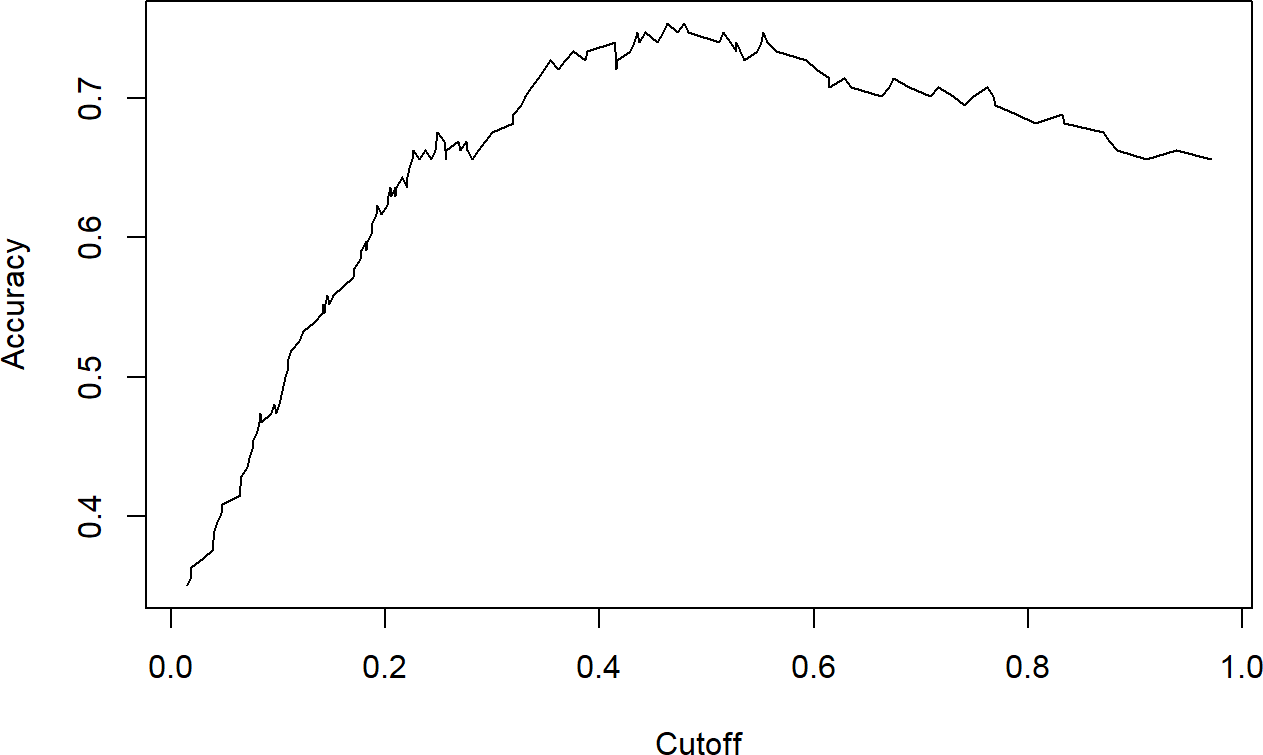
Source: Figure by author(s).

**Figure 65**. Sensitivity, specificity test data.

*# Accuracy (all correct / all) by cutoff:*

perf <- performance(pred,"acc","cutoff")

plot(perf)



Source: Figure by author(s).

**Figure 66**. Accuracy by cutoff test data.

*# Re-create the performance object for Sensitivity and Specificity*

perf\_sens\_spec <- performance(pred, "sens", "spec")

*# The highest sensitivity + specificity*

[max(perf\_sens\_spec@x.values[[](mailto:max(perf_sens_spec@x.values)1]] + [perf\_sens\_spec@y.values[[](mailto:perf_sens_spec@y.values)1]])

## [1] 1.438148

*# The cutoff that yields the highest sensitivity + specificity*

[perf\_sens\_spec@alpha.values[[](mailto:perf_sens_spec@alpha.values)1]][[which.max(perf\_sens\_spec@x.values[[](mailto:which.max(perf_sens_spec@x.values)1]] +

[perf\_sens\_spec@y.values[[](mailto:perf_sens_spec@y.values)1]])]

## [1] 0.4147078

*# AUC*

perf\_auc <- performance(pred,"auc")

[perf\_auc@y.values[[](mailto:perf_auc@y.values)1]]

## [1] 0.7809259

Our model performance on *testData* is quite similar to model performance on training data. Specifically, the model achieved an AUC of 0.81 on the training data, and an AUC of 0.78 on the test data. These two values are very close, which is an excellent result. It means our model is not overfit and generalizes well to new data, making it a reliable predictive model.

## Useful Resources

* “ROCR” package vignette [40].
* [Example diabetes data](https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv) (https://raw.githubusercontent.com/plotly/datasets/master/diabetes.csv").

## Chapter Summary

* Logistic Regression is used to predict categorical variables (model the logarithm of the probability of an outcome occurring)
* Logistic regression model may be fitted using the *glm()* function with *family*

*= binomial(link = "logit")*

* Maximum likelihood *logLik(), AIC()* for Akaike Information Criterion, *BIC()* for Bayesian Information Criterion, *pR2()* for McFadden’s pseudo R-squared helps select the logistic regression model that best fits the data
* Model evaluation may be done with the “*ROCR”* package (e.g., *prediction()* and

*performance()* functions)

* + Classification metrics: sensitivity, specificity, accuracy,. . .
  + Discrimination power: ROC curve, AUC
  + Model Fit Statistics: AIC, BIC, Pseudo R-squared,. . .
* Model validation: fit the model on training data and validate on test data

# Survival Data Analysis with R

This chapter introduces the essential concepts and practical application of survival data analysis in R, covering key statistical concepts like the Survival Function, Hazard Function, Censoring,… The practical implementation centers on the R packages “*survival”* and “*survminer”*. Readers will learn how to create and visualize non-parametric Kaplan-Meier survival curves, and how to test for differences between survival curves. The main analytical focus is the Cox Proportional Hazards Model to estimate Hazard Ratios (HR) while adjusting for covariates. The chapter also covers essential model validation steps, including checking the Proportional Hazards Assumption and more advanced topics such as Competing Risk Analysis and handling Time Dependent Covariates.

## Introduction to Survival Analysis

Survival analysis is the study of survival times and of the factors that influence them. Survival analysis investigates the time it takes for an event of interest to occur (time-to-event).

In many biomedical and clinical studies, the outcome of interest is time until an event occurs, for example:

* Time from birth until death
* Time from entry into a clinical trial until tumor response
* Time from treatment initiation to disease recovery

And much more. . .

Below are some other application of survival analysis:

* Reliability analysis
* Duration analysis
* Event history analysis
* Time-to-event analysis

### Survival Data

* + - * Time to event: response variable is a non-negative discrete or continuous random variable, and represents the time from a well-defined origin (e.g., the date of diagnosis, the start of treatment, or birth. . . ) to a well-defined event (e.g., death, recovery. . . ) (distinct start time and end time).
      * Survival data is often represented as a pair (*ti*; *δi*), where t is the time until

endpoint or last follow-up, and *δ* is a 0/1 variable with 0 = “subject was censored at t” and 1 = “subject had an event at t”, or in R code as Surv(time, status) [42].

* + - * Censoring: This occurs when the event of interest is not observed for a subject.

The most common form is right-censoring, where we know that the event has not happened by a certain time, but we do not know when it will happen. For example, a patient might still be alive at the end of a clinical trial but we don’t know when the patient might die.

### Key Statistical Concepts

* + - * Survival Function (*S*(*t*)): This function gives the probability that a subject will survive longer than a specified time t. It is a non-increasing function, starting at 1 at time 0 and approaching 0 as time increases.
      * Hazard Function (*h*(*t*)): Also known as the hazard rate or instantaneous

failure rate, the hazard function represents the instantaneous risk of the event

occurring at time t, given that the subject has survived up to that time [43].

## Essential R Packages for Survival Analysis

* *“survival”*: This is the essential core package for survival analysis in R. It contains functions for creating survival objects, fitting Kaplan-Meier curves, and building Cox proportional hazards models [43].
* *“survminer”:* This package provides functions for creating more informative and publication-ready visualizations of survival analysis results [44], building upon the “*ggplot2”* package [18].
* *“lubridate”:* This package is for working with dates (to create time to event data)

[45].

Packages "*glmnet*" for LASSO variable selection [46], and "*cmprsk*" for competing risk analysis [47] will be also used. Some supporting packages (*"tidyverse" [11], "knitr" [5], "kableExtra" [12]*) are routinely used as mentioned in previous chapters. First, we load the packages used in this chapter.

*#install.packages(c("survival","survminer","lubridate"))*

*#load multiple packages*

lapply(c("survival", "survminer", "lubridate",

"glmnet", "cmprsk", "tidyverse", "knitr", "kableExtra"), library, character.only = TRUE)

### Working with Dates

Data will often come with start and end dates rather than pre-calculated survival times.

* + - * Make sure these dates are formatted as dates in R.
      * Calculate survival time (time-to-event)

We first load the example data and convert dates to data using *as.Date()*. We then can calculate the number of days between the two dates using *difftime()* and convert it to a numeric value, then convert it to month by dividing by 30.42. Or, we can use “*lubridate*” package for this with *interval()*.

url\_base<-"https://stats.idre.ucla.edu/stat/r/examples/asa/" data\_use<-"hmohiv.csv"

exdate<-read.table(paste0(url\_base, data\_use), sep = ",", header = TRUE)

*#dates with base R*

*# calculate the number of days between our two dates # and convert it to a numeric value using as.numeric. # Then convert to month by dividing by 30.42.* exdateb<-exdate %>%

mutate\_at(c("entdate","enddate"), as.Date, format = "%m/%d/%Y")%>% mutate(

ttevent.m = as.numeric( difftime(enddate,

entdate,

units = "days"))/30.42)

head(exdateb)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | ID | time | age | drug | censor | entdate | enddate | ttevent.m |
| ## | 1 1 | 5 | 46 | 0 | 1 | 1990-05-15 | 1990-10-14 | 4.9967127 |
| ## | 2 2 | 6 | 35 | 1 | 0 | 1989-09-19 | 1990-03-20 | 5.9829060 |
| ## | 3 3 | 8 | 30 | 1 | 1 | 1991-04-21 | 1991-12-20 | 7.9881657 |
| ## | 4 4 | 3 | 30 | 1 | 1 | 1991-01-03 | 1991-04-04 | 2.9914530 |
| ## | 5 5 | 22 | 36 | 0 | 1 | 1989-09-18 | 1991-07-19 | 21.9921105 |
| ## | 6 6 | 1 | 32 | 1 | 0 | 1991-03-18 | 1991-04-17 | 0.9861933 |

*#with lubridate*

exdatel<-exdate%>%

mutate\_at(c("entdate","enddate"), mdy)%>%

mutate(

ttevent.m = interval(entdate,enddate)/months(1))

head(exdatel)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | ID | time | age | drug | censor | entdate | enddate | ttevent.m |
| ## | 1 1 | 5 | 46 | 0 | 1 | 1990-05-15 | 1990-10-14 | 4.9666667 |
| ## | 2 2 | 6 | 35 | 1 | 0 | 1989-09-19 | 1990-03-20 | 6.0046729 |
| ## | 3 3 | 8 | 30 | 1 | 1 | 1991-04-21 | 1991-12-20 | 7.9666667 |
| ## | 4 4 | 3 | 30 | 1 | 1 | 1991-01-03 | 1991-04-04 | 3.0081967 |
| ## | 5 5 | 22 | 36 | 0 | 1 | 1989-09-18 | 1991-07-19 | 22.0014265 |
| ## | 6 6 | 1 | 32 | 1 | 0 | 1991-03-18 | 1991-04-17 | 0.9677419 |

*#rename*

exdatel<-exdatel%>%

rename(status = censor)

### Package “Survival” Main Functions

Below are the basic functions of the “*survival*” package [43]:

*Surv()*

This function is for creating a survival object.

* + - * *Surv(time, status)*: right censored data
      * *Surv(time, endpoint = = “death”)*: right censored data, where the status variable

is a character or factor

* + - * *Surv(t1, t2, status)*: counting process data
      * *Surv(t1, ind, type = ‘left’)*: left censoring
      * *Surv(time, fstat)*: multiple state data, fstat is a factor

*coxph()*

This function is for fitting Cox’s Proportional Hazards Model

* + - * *coxph(Surv(time, status) ~ x, data = aml)*: standard Cox model
      * *coxph(Surv(t1, t2, stat) ~ (age + surgery)\* transplant)*: time dependent

covariates.

* + - * *y <- Surv(t1, t2, stat) coxph(y~strata(inst) \* sex + age + treat)*: Stratified model,

with a separate baseline per institution, and institution specific effects for sex.

* + - * *coxph(y ~ offset(x1) + x2)*: force in a known term, without estimating a

coeffcient for it.

*cox.zph()*

This function is for computing a test of proportional hazards for the fitted Cox model.

* + - * *zfit <- cox.zph(coxfit); plot(zfit)*.

*Survreg()*

This function is for fitting parametric survival models.

* + - * *survreg(Surv(time, stat) ~x, dist = “loglogistic”)*: Fit a log-logistic distribution.

*Survfit()*

This function is for fitting a Survival Curve.

* + - * *survfit(Surv(time, status))*: Simple Kaplan-Meier
      * *survfit(Surv(time, status) ~ rx + sex)*: Four groups
      * *fit <- coxph(Surv(time, stat) ~ rx + sex); survfit(fit, list(rx = 1, sex = 2))*: Predict

the curve

*survdiff()*

This function is for testing the difference between survival curves. This is one and k-sample versions of the Fleming-Harrington G family. It includes the logrank and Gehan-Wilcoxon as special cases.

* + - * *survdiff(Surv(time, status) ~ sex + treat)* : Compare the 4 sub-groups formed by sex and treatment combinations.
      * *survdiff(Surv(time, status) ~ offset(pred)):* One-sample test

## Survival Curves and Models

### Survival Model and Curve for Overall Data

The codes below fit basic survival curve. In which, *Surv(time, event)* creates a special survival object, *survfit()* calculates survival probabilities at every event time while handling censored observations, ~ 1 specifies the overall survival curve (i.e., no stratification by covariates).

fit1 <- survfit(Surv(time, status) ~ 1, data = lung)

print(fit1)

*#summary(fit1)*

## Call: survfit(formula = Surv(time, status) ~ 1, data = lung) ##

## n events median 0.95LCL 0.95UCL

## [1,] 228 165 310 285 363

The output shows:

* Median survival time and 95%CI: the time at which survival probability drops to 0.50.
* The number at risk, the number of events

What if one calculates median survival time as below? Is it right or wrong and why?

lung %>% filter(status == 1)%>% summarise(med\_surv = median(time))

## med\_surv ## 1 284

This is wrong as censoring is ignored. The correct estimate is the median time in the output from the *survfit()* object:

print(fit1)

## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)

##

## n events median 0.95LCL 0.95UCL

## [1,] 228 165 310 285 363

To calculate survival as a specific time point, for example survival at 1 year, should one do like below?

evy1<-lung %>%

filter(time< = 365 & status == 1) %>%

summarise(

sum(as.numeric(as.character(status))))

1-evy1/nrow(lung)

## sum(as.numeric(as.character(status)))

## 1 0.8157895

The above is wrong as censoring is ignored. The correct estimate for 1 year survival is as below when censoring is taken into account. The function *summary(survfit object)* displays survival probabilities at each event time (or at one specific selected time point) and median survival estimates.

summary(fit1, times = 365) ## Call: survfit(formula = Surv(time, status) ~ 1, data = lung)

##

## time n.risk n.event survival std.err lower 95% CI upper 95% CI

## 365 65 121 0.409 0.0358 0.345 0.486

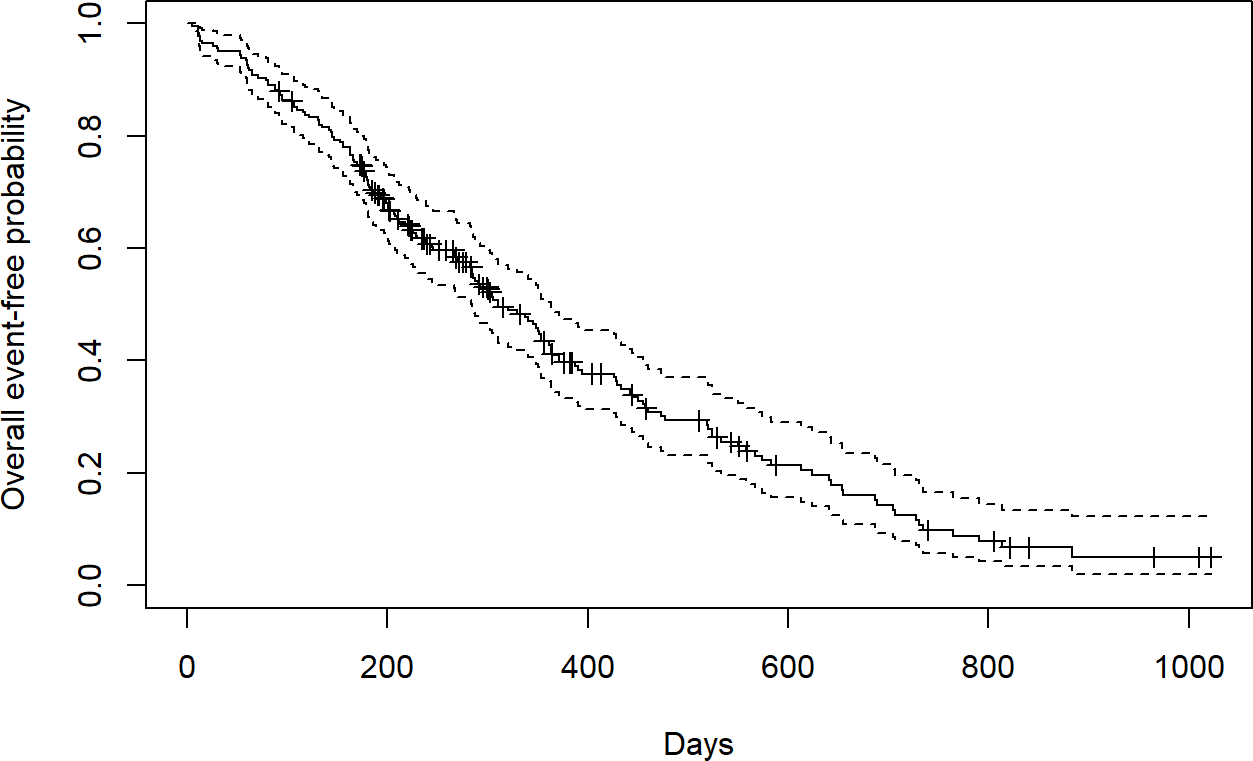
#### Kaplan-Meier Estimator

The Kaplan-Meier (KM) estimator is a non-parametric statistic used to estimate the survival function from time-to-event data. The y-axis of the KM plot shows the probability of surviving (or being event-free) over time (x-axis), demonstrating the decrease in survival probability over the follow-up period. Create a Kaplan-Meier plot using base R:

plot(fit1, mark.time = TRUE,

xlab = "Days",

ylab = "Overall event-free probability")



Source: Figure by author(s).

**Figure 67**. Kaplan-Meier plot using base R.

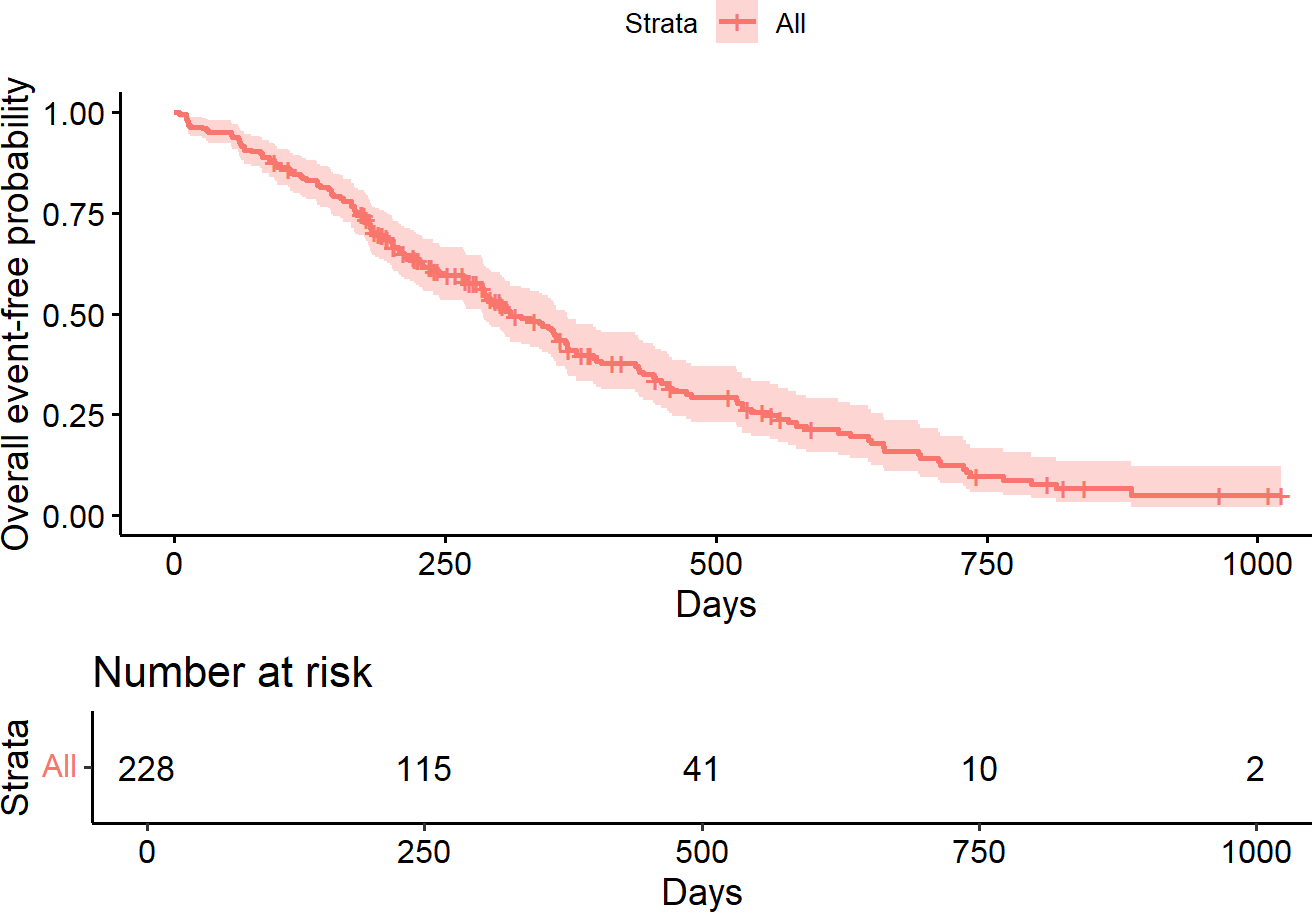
Kaplan-Meier plot may be better visualized using the “*survminer”* package and its *ggsurvplot()* function. There are many options that can be used to adjust the display of the Kaplan-Meier plot (see *help (ggsurvplot, survminer)*). For example, option *risk.table = TRUE* adds the number of subjects at risk over time below the main plot.

ggsurvplot(fit = fit1,

risk.table = TRUE, risk.table.height = 0.3,

xlab = "Days",

ylab = "Overall event-free probability")



Source: Figure by author(s).

**Figure 68**. Kaplan-Meier plot using “survminer” package.

### Comparing Survival Times between Groups

By changing the above formula from *~ 1* to *~ sex*, *survfit()* calculates separate KM estimates for each level of the sex variable (Male=1, Female=2).

fit2 <- survfit(Surv(time, status) ~ sex, data = lung)

print(fit2)

*#summary(fit2)*

## Call: survfit(formula = Surv(time, status) ~ sex, data = lung) ##

## n events median 0.95LCL 0.95UCL

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | sex = | 1 138 | 112 | 270 | 212 | 310 |
| ## | sex = | 2 90 | 53 | 426 | 348 | 550 |

The output shows that the estimated median survival time is 270 days for Male (*sex=1*) and 426 days for Female (*sex=2*), suggesting a large difference in survival time in favor of females.

#### Testing the Difference

The most common way to compare survival times between groups is log-rank test (or Mantel-Haenszel test) which equally weights observations over the entire follow-up time. (rho = 0 for log rank test (default); rho = 1 for Peto-Wilcoxon test).

The log-rank test is equivalent to the Score test from a Cox model with the group as a factor variable.

survdiff(Surv(time, status) ~ sex, data = lung)

## Call:

## survdiff(formula = Surv(time, status) ~ sex, data = lung)

##

## N Observed Expected (O-E)^2/E (O-E)^2/V

## sex = 1 138 112 91.6 4.55 10.3

## sex = 2 90 53 73.4 5.68 10.3

##

## Chisq = 10.3 on 1 degrees of freedom, p = 0.001

survdiff(Surv(time, status) ~ sex, rho = 1, data = lung)

## Call:

## survdiff(formula = Surv(time, status) ~ sex, data = lung,

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rho = | | 1) |  | | | | | |
| ## | |  |
| ## | |  | N Observed Expected (O-E)^2/E (O-E)^2/V | | | | | |
| ## | sex | = 1 | 138 | 70.4 | 55.6 | 3.95 | 12.7 | |
| ## | sex | = 2 | 90 | 28.7 | 43.5 | 5.04 | 12.7 | |
| ## |  |  |  |  |  |  |  | |
| ## | Chisq = 12.7 | | | on 1 | degrees of | freedom, | p = | 0.0004 |

The output shows that the p-value of 0.001 is highly significant and we may conclude that there is a statistically significant difference in the survival distributions between males and females.

The Kaplan-Meier curve between groups and the p-value comparing between group may be visualized using the “*survminer*” packages as below:

ggsurvplot( fit = fit2,

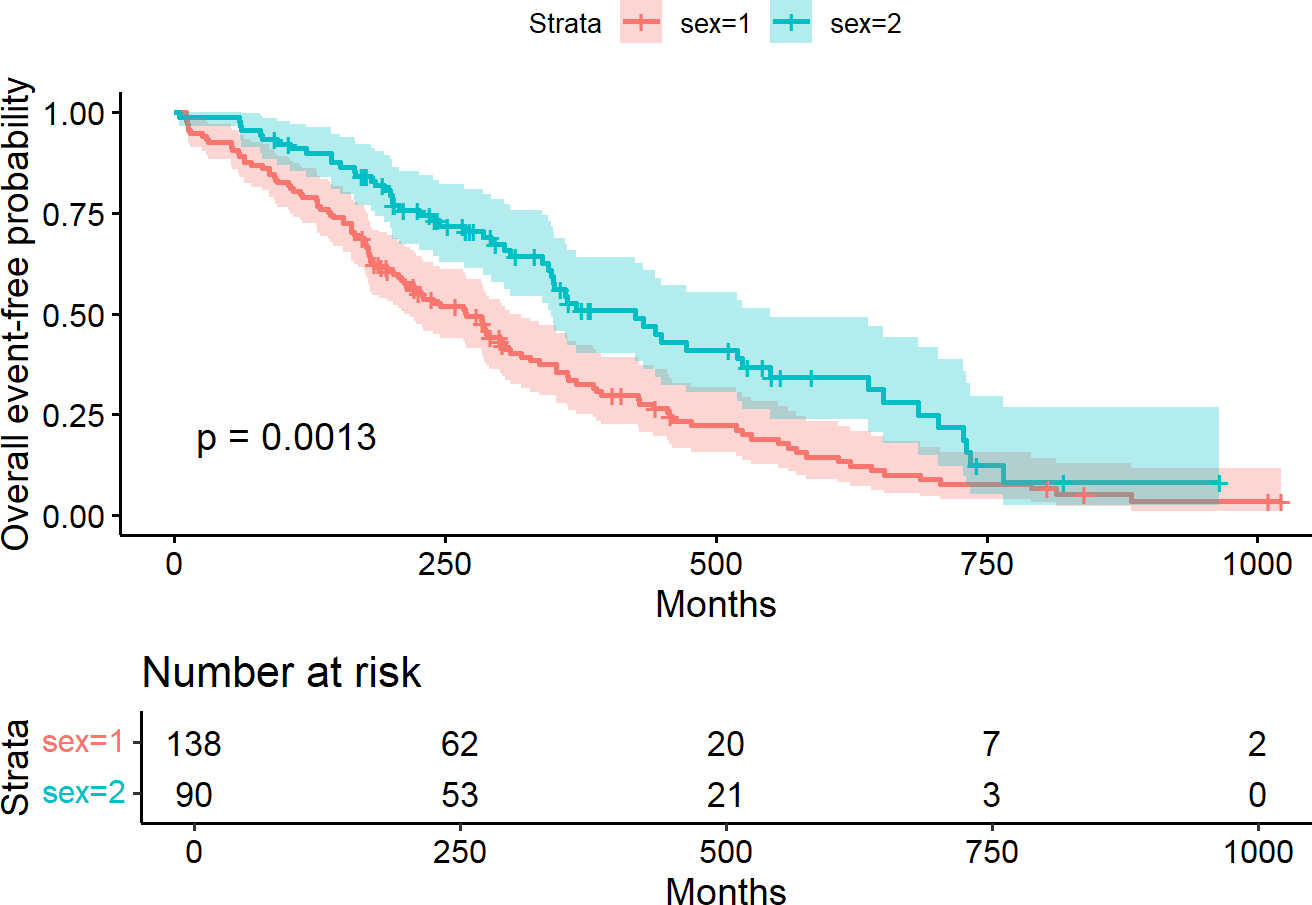
conf.int = TRUE,

pval = TRUE,

risk.table = TRUE,

risk.table.height = 0.3, xlab = "Months",

ylab = "Overall event-free probability")



Source: Figure by author(s).

**Figure 69**. Kaplan-Meier curve between groups with “survminer”.

#### The Cox Proportional Hazards Model

The Cox proportional hazards model is a semi-parametric model that is used to investigate the relationship between survival outcome and one or more covariates. Some key assumptions of the model are: non-informative censoring and proportional hazards (the effect of the covariates on the hazard is constant over time) [48].

The Cox model is expressed as: *h*(*t*|*X*) = *h*0(*t*) × exp(*β*1*X*1 + *β*2*X*2 + · · · +

*βpXp*)

Where:

* *h*(*t*|*X*) is the hazard rate at time t for an individual with covariates X.
* *h*0(*t*) is the baseline hazard function, which is the hazard when all covariates are zero.
* *exp*(*βp*) is the hazard ratio (HR) associated with the covariate *Xp [38]*.

The Cox model is fitted with *coxph()*, and summarized with *summary(Cox model object)* which reports coefficients (β), HR = exp(β), standard error, z-statistic, p-value, ….

cfit1 <- coxph(Surv(time, status) ~ sex, data = lung)

print(cfit1, digits = 3)

## Call:

## coxph(formula = Surv(time, status) ~ sex, data = lung)

##

## coef exp(coef) se(coef) z p

## sex -0.531 0.588 0.167 -3.18 0.0015

##

## Likelihood ratio test = 10.6 on 1 df, p = 0.00111

## n = 228, number of events = 165

*# print routine gives a short summary and # the summary routine a longer one*

summary(cfit1, digits = 3)

## Call:

## coxph(formula = Surv(time, status) ~ sex, data = lung) ##

## n = 228, number of events = 165 ##

## coef exp(coef) se(coef) z Pr(>|z|)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | sex | -0.5310 | 0.5880 | 0.1672 | -3.176 | 0.00149 \*\* |
| ## | --- |  |  |  |  |  |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## exp(coef) exp(-coef) lower .95 upper .95

## sex 0.588 1.701 0.4237 0.816

##

## Concordance = 0.579 (se = 0.021 )

## Likelihood ratio test = 10.63 on 1 df, p = 0.001

## Wald test = 10.09 on 1 df, p = 0.001

## Score (logrank) test = 10.33 on 1 df, p = 0.001

Likelihood ratio test for the significance of a variable (e.g. “sex) may be done as below. First, fit the Cox model without “sex” *cfit0*, then fit the model with “sex” *cfit1*, then use the function *anova(cift1, cfit0)* or simply *anova(cift1)* to compare the two models *cfit1* and *cfit0*.

*#likelihood ratio test*

anova(cfit1)

## Analysis of Deviance Table

## Cox model: response is Surv(time, status)

## Terms added sequentially (first to last)

##

## loglik Chisq Df Pr(>|Chi|)

## NULL -749.91

## sex -744.59 10.634 1 0.001111 \*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

cfit0 <- coxph(Surv(time, status) ~ 1,data = lung)

anova(cfit1,cfit0)

## Analysis of Deviance Table

## Cox model: response is Surv(time, status)

## Model 1: ~ sex

## Model 2: ~ 1

## loglik Chisq Df Pr(>|Chi|)

## 1 -744.59

## 2 -749.91 10.634 1 0.001111 \*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

### Hazard ratios

* Hazard ratio (HR) represents the ratio of hazards between two groups at any particular point in time.
* Interpreted as the instantaneous rate of occurrence of the event of interest in those who are still at risk for the event. It is not a risk, though it is commonly interpreted as such
* *HR* = *exp*(*β*) (exponentiation of coefficients of Cox model output)
* A HR < 1 indicates reduced hazard of event whereas a HR > 1 indicates an

increased hazard of event.

The output shows that HR is 0.59 which implies that the ratio of the hazard of event of sex = 2 over the hazard of event of sex = 1 is 0.59.

### 

Multivariable Cox model may be done by adding covariates to the Cox model, for example “sex” and “age”.

*# covariates*

cfit2 <- coxph(Surv(time, status) ~ sex + age, data = lung)

print(cfit2, digits = 3)

## Call:

## coxph(formula = Surv(time, status) ~ sex + age, data = lung) ##

## coef exp(coef) se(coef) z p ## sex -0.51322 0.59857 0.16746 -3.06 0.0022

## age 0.01705 1.01719 0.00922 1.85 0.0646

##

## Likelihood ratio test = 14.1 on 2 df, p = 0.000857

## n = 228, number of events = 165

summary(cfit2, digits = 3)

|  |  |
| --- | --- |
| ##  ## | Call:  coxph(formula = Surv(time, status) ~ sex + age, data = lung) |
| ## |  |
| ## | n = 228, number of events = 165 |
| ## |  |
| ## | coef exp(coef) se(coef) z Pr(>|z|) |
| ## | sex -0.513219 0.598566 0.167458 -3.065 0.00218 \*\* |
| ## | age 0.017045 1.017191 0.009223 1.848 0.06459 . |
| ## | --- |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 |
| ## | exp(coef) exp(-coef) lower .95 upper .95 |
| ## | sex 0.5986 1.6707 0.4311 0.8311 |
| ## | age 1.0172 0.9831 0.9990 1.0357 |
| ## |  |
| ## | Concordance = 0.603 (se = 0.025 ) |
| ## | Likelihood ratio test = 14.12 on 2 df, p = 0.0009 |
| ## | Wald test = 13.47 on 2 df, p = 0.001 |
| ## | Score (logrank) test = 13.72 on 2 df, p = 0.001 |

The output shows the HR (*exp(coef)*) and 95%CI of “sex” and “age”.

### Stratified Cox model

* Adding one or more strata terms to the model formula.
* Each subject is compared only to subjects within their own stratum for

computing the partial likelihood, and then the final results are summed over the strata.

* Variable included as a stratum is adjusted for in the most general way, and not

having an estimate of its effect.

* Common use of strata: to adjust for the enrolling institution in a multi-center study.

The below example adds *strata()* to the Cox model and compare the coefficients of the Cox models with and without *strata()*.

*# Get complete dataset with no missing values*

lung\_complete <- na.omit(lung)

*# Example*

cfit <- coxph(Surv(time, status) ~ age + sex + wt.loss,

data = lung\_complete)

*#summary(cfit)*

cfits <- coxph(Surv(time, status) ~ age + sex + wt.loss + strata(inst),

data = lung\_complete)

*#summary(cfits)*

*# compare coefficients between simple and stratified models*

round(cbind(simple = coef(cfit), stratified = coef(cfits)), 4)

|  |  |  |  |
| --- | --- | --- | --- |
| ## |  | simple | stratified |
| ## | age | 0.0174 | 0.0229 |
| ## | sex | -0.4491 | -0.5346 |
| ## | wt.loss | -0.0012 | -0.0034 |

## Model Evaluation

### Checking Assumptions for Cox Model

Cox proportional hazards models should be checked for the following assumptions:

* + - * Proportional hazards
      * Additivity
      * Linearity

#### Checking Proportional Hazards Assumption

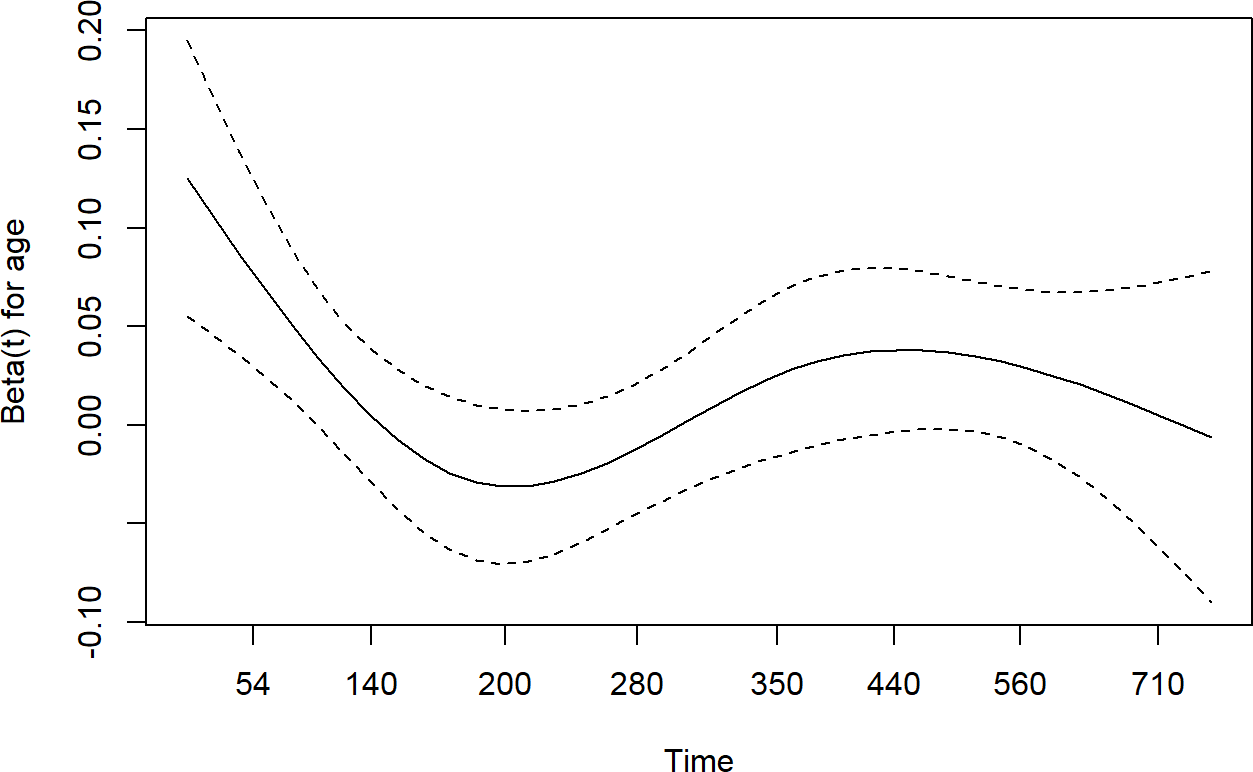
Checking Proportional Hazards Assumption may be done using *cox.zph(Cox model fit)* and plot the results.

zp1 <- cox.zph(cfit)

zp1

|  |  |  |  |
| --- | --- | --- | --- |
| ## | chisq | df | p |
| ## age | 0.668 | 1 | 0.41 |
| ## sex | 1.003 | 1 | 0.32 |
| ## wt.loss | 0.351 | 1 | 0.55 |
| ## GLOBAL | 1.868 | 3 | 0.60 |

plot(zp1[1], resid = FALSE)



Source: Figure by author(s).

**Figure 70**. Check Proportional hazards.

The output shows a test for each covariate and a global test. All p-values are not significant, thus we cannot reject proportional hazards assumption (the assumption is met).

#### Checking Additivity (no interaction)

This is often checked by testing for a significant interaction term.

cfitm<-update(cfit, . ~ . + age\*sex)

anova(cfit, cfitm)

## Analysis of Deviance Table

## Cox model: response is Surv(time, status)

## Model 1: ~ age + sex + wt.loss

## Model 2: ~ age + sex + wt.loss + age:sex

## loglik Chisq Df Pr(>|Chi|)

## 1 -503.66

## 2 -503.54 0.238 1 0.6257

The output shows that the interaction term is not significant thus, additivity assumption cannot be rejected.

#### Checking Linearity

For continuous covariates like “age”, the Cox model assumes a linear relationship between the covariate and the log-hazard. This is checked using restricted cubic splines (via *pspline()*) and the *termplot()* function). In the example below, *pspline(age)* models age non-linearly, *termplot()* plots the estimated effect of age on the log-hazard.

cfitl<-coxph(Surv(time, status) ~ pspline(age) + sex + wt.loss,

lung\_complete)

print(cfitl, digit = 3)

## Call:

## coxph(formula = Surv(time, status) ~ pspline(age) + sex + wt.loss,

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | data = lung\_complete) | | |  | | | | |
| ## |  | | |
| ## | coef se(coef) | | | se2 | | | Chisq | DF |
| ## | pspline(age), linear 0.01611 0.01061 | | | 0.01061 | | | 2.30460 | 1.00 |
| ## | pspline(age), nonlin | | |  | | | 4.92319 | 3.05 |
| ## | sex -0.49026 0.20007 | | | 0.19958 | | | 6.00493 | 1.00 |
| ## | wt.loss -0.00129 0.00678 | | | 0.00676 | | | 0.03621 | 1.00 |
| ## | p | | |  | | |  |  |
| ## | pspline(age), linear 0.129 | | |  | | |  |  |
| ## | pspline(age), nonlin 0.183 | | |  | | |  |  |
| ## | sex 0.014 | | |  | | |  |  |
| ## | wt.loss 0.849 | | |  | | |  |  |
| ## |  | | |  | | |  |  |
| ## | Iterations: 4 outer, 12 Newton-Raphson | | |  | | |  |  |
| ## | Theta = 0.751 | | |  | | |  |  |
| ## | Degrees of freedom for terms = 4 1 1 | | |  | | |  |  |
| ## | Likelihood ratio | test = 15.4 | on 6.04 | df, p | = | 0.02 | | |
| ## | n = 167, number | of events = | 120 |  |  |  | | |

*#Plot effect of age (quite linear)*

termplot(cfitl, term = 1, se = TRUE)



Source: Figure by author(s).

**Figure 71**. Check linearity.

The output shows that non-linearity (p = 0.183) is not significant. The figure shows the estimated effect (solid line) and its 95% confidence bounds (dashed lines). If the solid line is mostly linear and the confidence band is narrow and covers the linear trend, the linearity assumption is met.

### Evaluating Model Accuracy

A good model should accurately discriminate between individuals with different risks. The most common measure of this discriminatory power for survival models is the Concordance Index (C-index). It is analogous to the Area Under the Curve (AUC) in logistic regression. As a common rule, a C-index of 1.0 indicates perfect prediction, a C-index of 0.5 indicates that the model is no better than random chance and a C-index > 0.7 is generally considered to indicate a reasonably good model.

The output of *summary(Cox model)* displays C-index:

lung\_complete<-na.omit(lung)

cfit <- coxph(Surv(time, status) ~ age + sex + wt.loss,

data = lung\_complete)

summary(cfit)

## Call:

## coxph(formula = Surv(time, status) ~ age + sex + wt.loss, data = lung\_complete)

##

## n = 167, number of events = 120 ##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | coef | exp(coef) | se(coef) | z | Pr(>|z|) |
| ## age | 0.017400 | 1.017553 | 0.010836 | 1.606 | 0.1083 |
| ## sex | -0.449121 | 0.638189 | 0.197983 | -2.268 | 0.0233 \* |
| ## wt.loss | -0.001213 | 0.998788 | 0.006823 | -0.178 | 0.8589 |
| ## --- |  |  |  |  |  |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | exp(coef) | exp(-coef) | | lower .95 | upper .95 |
| ## age | 1.0176 | 0.9828 | | 0.9962 | 1.0394 |
| ## sex | 0.6382 | 1.5669 | | 0.4329 | 0.9407 |
| ## wt.loss | 0.9988 | 1.0012 | | 0.9855 | 1.0122 |
| ## |  |  | |  |  |
| ## Concordance = 0.602 | | | (se = | 0.032 ) |  |
| ## Likelihood ratio test | | | = 8.91 | on 3 df, | p = 0.03 |
| ## Wald test | | | = 8.5 | on 3 df, | p = 0.04 |
| ## Score (logrank) test | | | = 8.62 | on 3 df, | p = 0.03 |

The output shows that the C-index is 0.6016187 suggesting the model performs slightly better than random chance. A value above 0.7 is generally considered to indicate a reasonably good model.

### Residual Analysis for Goodness-of-Fit

Residuals are used to identify potential outliers and non-linear patterns.

* Martingale Residuals: Used to check the functional form (linearity) of continuous covariates. If plotted against a covariate (e.g., Age), a non-random pattern suggests the need for a non-linear term (like a spline or polynomial) or data transformation.
* Deviance Residuals: Used to identify subjects whose observed survival time is unusually long or short compared to the model's prediction (i.e., potential outliers).

We can extract Martingale and Deviance residuals using *residuals()*:

lung\_complete$martingale\_res <- residuals(cfit, type = "martingale")

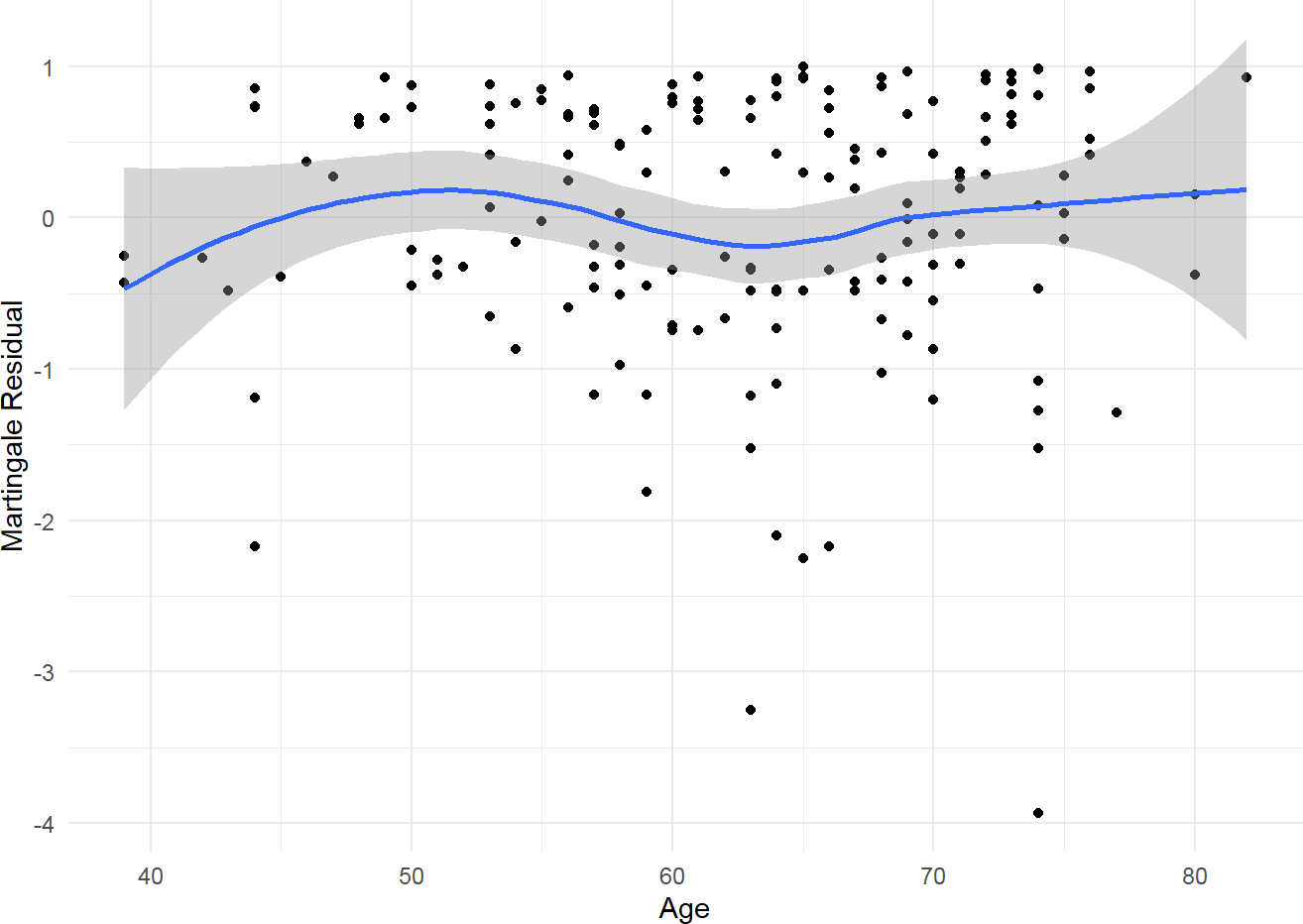
lung\_complete$deviance\_res <- residuals(cfit, type = "deviance")

The example below plots Martingale residuals against a continuous predictor (age) to check for non-linear patterns: *residuals(cfit, type = "martingale")* extracts the martingale residuals, *geom\_smooth(method = "loess")* plots a smoothed line of the relationship.

ggplot(lung\_complete, aes(x = age, y = martingale\_res)) + geom\_point() +

geom\_smooth(method = "loess") +

labs(x = "Age", y = "Martingale Residual") + theme\_minimal()

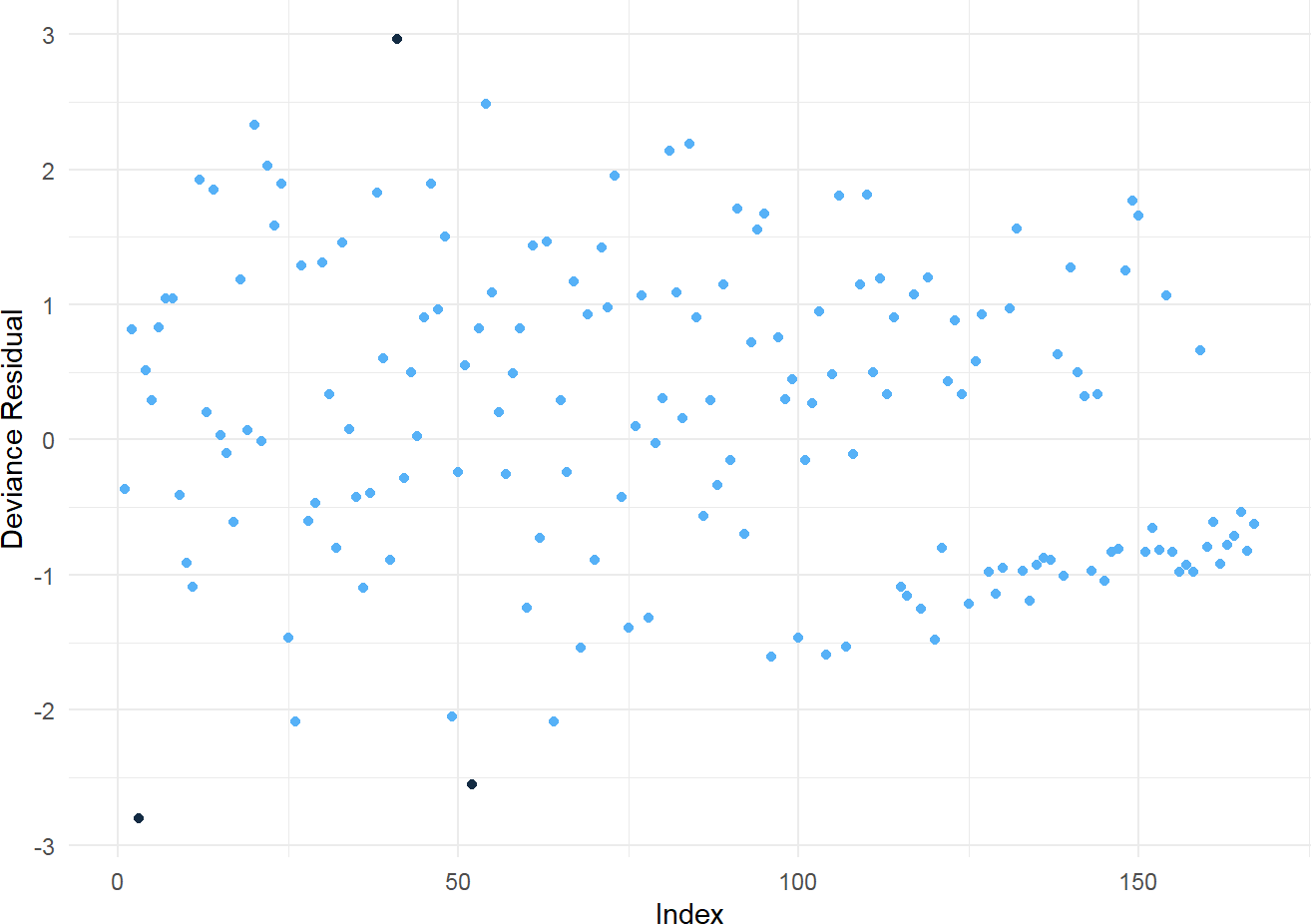


Source: Figure by author(s).

**Figure 72**. Martingale Residuals vs. Age.

The above plot shows the Martingale Residuals vs. “Age”. A horizontal smoothed line (around 0) suggests the functional form for “Age” is correct. If the line bends, it suggests non-linearity.

The example below plots Deviance residuals to identify potential outliers (in black):



ggplot(lung\_complete, aes(x = 1:nrow(lung\_complete),

y = deviance\_res)) +

geom\_point(aes(color

labs(x = "Index", y theme\_minimal() +

theme(legend.position = "none")

= ifelse(abs(deviance\_res) > 2.5, 1, 2))) +

= "Deviance Residual") +

Source: Figure by author(s).

**Figure 73**. Deviance Residuals.

## Model Selection

### Partial Likelihood Hypothesis Tests

We can use three forms of test for *β*: H0: *β* = 0:t

* + - * Wald test: commonly used test
      * Score test: equivalent to the log-rank test
      * Likelihood ratio test: key advantage of this test over the other two is that it is invariant to monotonic transformations of *β*.

Likelihood ratio tests may be done using *anova()* function:

anova(cfit)

## Analysis of Deviance Table

## Cox model: response is Surv(time, status)

## Terms added sequentially (first to last)

##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | loglik | Chisq | Df | Pr(>|Chi|) |  |
| ## NULL | -508.12 |  |  |  |
| ## age | -506.36 | 3.5236 | 1 | 0.06050 | . |
| ## sex | -503.68 | 5.3571 | 1 | 0.02064 | \* |
| ## wt.loss | -503.66 | 0.0318 | 1 | 0.85839 |  |
| ## --- |  |  |  |  |  |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The *anova()* function shows likelihood ratio tests for each term in a model, added sequentially (compared to the null model *cfit0*). The output shows that the p-value for “sex” is significant.

### Information Criteria

The Akaike Information Criterion (AIC) with *AIC()* and Bayesian Information Criterion (BIC) with *BIC()* may be used to compare Non-nested Models. A lower value indicates a better model.

AIC(cfit)

## [1] 1013.321

AIC(cfits)

## [1] 401.7227

### Model Selection Approaches

With large number of potential factors, the model needs to be pruned so that only necessary covariates are included.

#### Stepwise Variable Selection

* + - * Backward Elimination: Starts with a “full” model containing all candidate predictors. In each step, it removes the predictor whose removal results in the largest improvement (lowest value) of the AIC. The process stops when no further removals improve the model.
      * Forward Selection: Starts with an “empty” model (intercept only). In each step,

it adds the predictor that provides the best improvement in the AIC.

* + - * Stepwise (Bidirectional): A combination of forward and backward steps. Stepwise model selection may be done using *step()* function:

lung\_complete<-na.omit(survival::lung)

fitall2<-coxph(Surv(time, status) ~ age + sex + wt.loss+ inst+ ph.ecog+ ph.karno+ meal.cal,

data = lung\_complete)

aicstep2<-step(fitall2, scope = list(upper = ~age +sex + wt.loss+ inst+

ph.ecog+ ph.karno+ meal.cal, lower = ~age))

## Start: AIC = 998.33

## Surv(time, status) ~ age + sex + wt.loss + inst + ph.ecog + ph.karno + ## meal.cal

##

## Df AIC

## - meal.cal 1 996.38

## <none> 998.33

## - wt.loss 1 1000.12

## - ph.karno 1 1001.23

## - inst 1 1002.27

## - sex 1 1005.05

## - ph.ecog 1 1015.67 ##

## Step: AIC = 996.38

## Surv(time, status) ~ age + sex + wt.loss + inst + ph.ecog + ph.karno ##

## Df AIC

## <none> 996.38

## - wt.loss 1 998.16

## + meal.cal 1 998.33

## - ph.karno 1 999.27

## - inst 1 1000.29

## - sex 1 1003.07

## - ph.ecog 1 1013.83

aicstep2

## Call:

## coxph(formula = Surv(time, status) ~ age + sex + wt.loss + inst +

## ph.ecog + ph.karno, data = lung\_complete)

##

## coef exp(coef) se(coef) z p

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | age | 0.013784 | 1.013880 | 0.011717 | 1.176 | 0.23943 |
| ## | sex | -0.572214 | 0.564275 | 0.199575 | -2.867 | 0.00414 |
| ## | wt.loss | -0.014682 | 0.985425 | 0.007765 | -1.891 | 0.05866 |
| ## | inst | -0.031333 | 0.969152 | 0.013087 | -2.394 | 0.01665 |
| ## | ph.ecog | 0.993069 | 2.699507 | 0.229849 | 4.321 | 0.0000156 |
| ## | ph.karno | 0.024618 | 1.024923 | 0.011472 | 2.146 | 0.03189 |
| ## |  |  |  |  |  |  |
| ## | Likelihood ratio test = 31.85 | | | on 6 df, | p = 0.0000174 | |
| ## | n = 167, number of events = | | | 120 |  | |

The output shows the final selected model after *Call*.

#### Penalized Regression (LASSO)

For situations with a very large number of predictors (high-dimensional data), traditional stepwise methods can be unstable. Penalized regression, particularly LASSO (Least Absolute Shrinkage and Selection Operator), is a more modern and robust alternative.

LASSO adds a penalty term to the model-fitting process that is proportional to the absolute value of the coefficients. This penalty forces the coefficients of less important variables to shrink towards, and often exactly to, zero. Therefore, LASSO performs both coefficient estimation and variable selection simultaneously. It is particularly effective at preventing overfitting.

LASSO approach may be done using *cv.glmnet()* the “*glmnet”* package [46]:

*#library(glmnet)*

*# glmnet requires a matrix of predictors (x) and a survival object (y)*

x\_vars <- model.matrix( ~ age + sex + wt.loss+ inst+ ph.ecog+

ph.karno+ meal.cal,

data = lung\_complete)[,-1]

y\_surv <- Surv(lung\_complete$time, lung\_complete$status)

*# Use cross-validation to find the optimal penalty parameter (lambda)*

cv\_lasso <- cv.glmnet(x\_vars, y\_surv, family = "cox", alpha = 1)

*# The optimal lambda that minimizes cross-validated error*

optimal\_lambda <- cv\_lasso$lambda.min

*# Extract the coefficients at the optimal lambda*

lasso\_coefs <- coef(cv\_lasso, s= optimal\_lambda)

print(lasso\_coefs)

## 7 x 1 sparse Matrix of class "dgCMatrix"

## 1

## age 0.0005103152

## sex -0.3236625195

|  |  |  |
| --- | --- | --- |
| ## | wt.loss | -0.0018539813 |
| ## | inst | -0.0088237865 |

|  |  |  |
| --- | --- | --- |
| ## | ph.ecog | 0.3846742971 |
| ## | ph.karno | . |
| ## | meal.cal | . |

The output shows that LASSO has eliminated “ph.karno” and “meal.cal” by shrinking their coefficients to exactly zero (.), and selected a model with “age”, “sex”, “wt.loss”, “inst”, and “ph.ecog”.

## Transform Survival Curves

Cumulative Events Function (Cumulative Incidence Function (CIF)), is a simple transformation of the Kaplan-Meier estimate: *F*(*t*) = 1 − *S*(*t*) Where *F*(*t*) is the cumulative probability of the event occurring at or before time t.

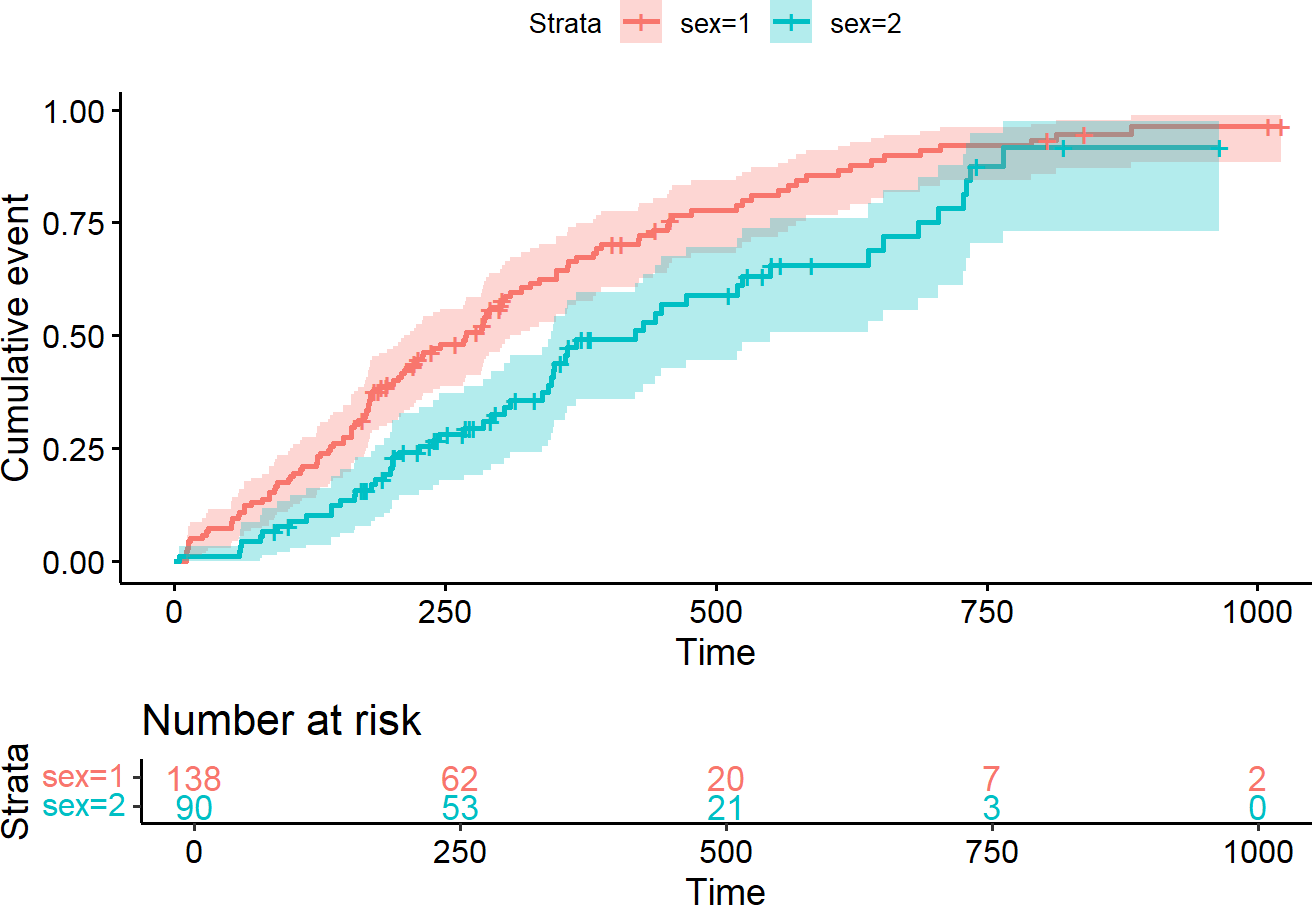
fit2 <- survfit(Surv(time, status) ~ sex, data = lung)

*#plot cumulative event (incidence)*

ggsurvplot(fit2, conf.int = TRUE,

fun = "event", *# Key argument for the CIF transformation*

risk.table = TRUE, risk.table.col = "strata")



Source: Figure by author(s).

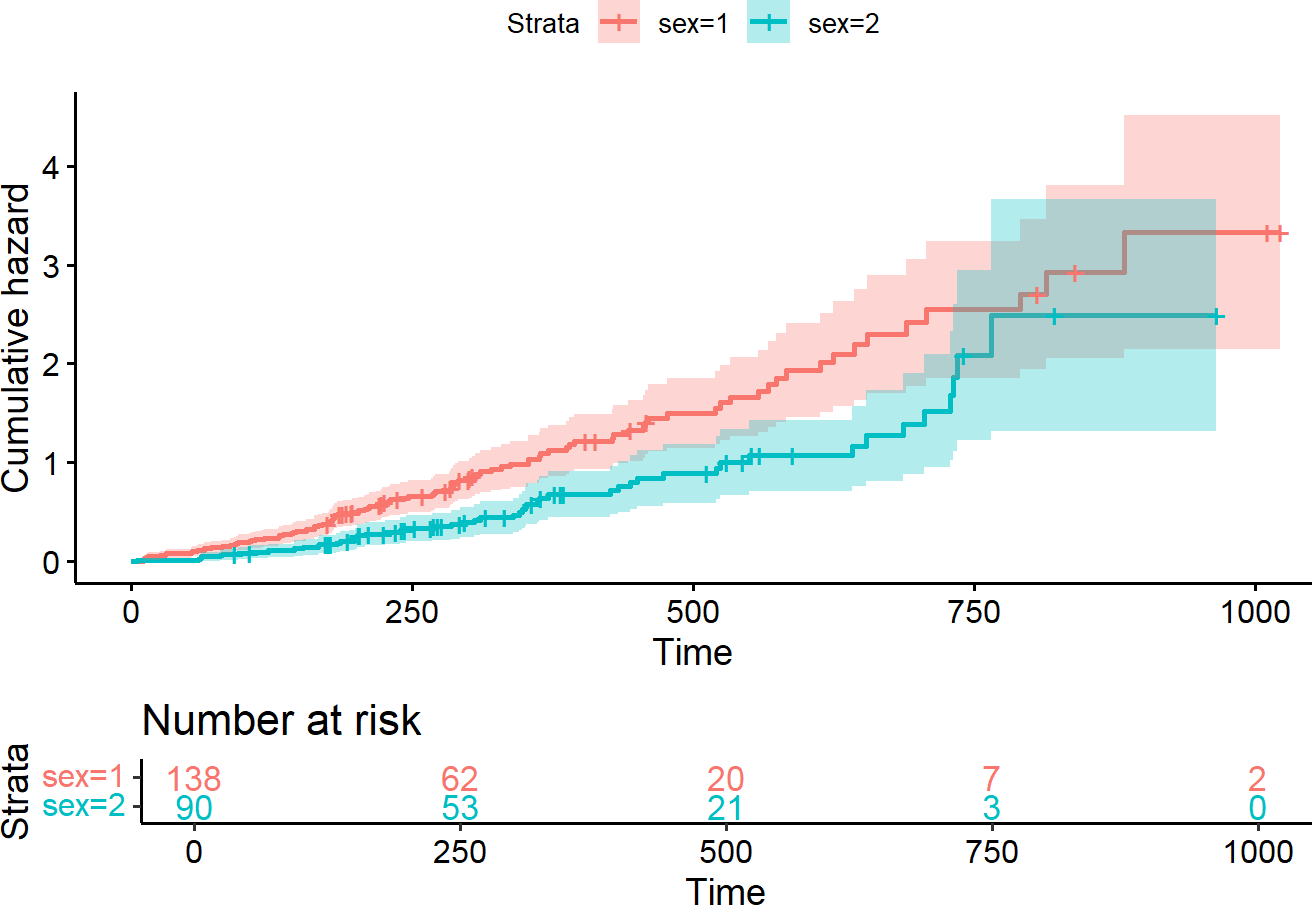
**Figure 74**. Cumulative Incidence Function.

Cumulative Hazard Function (CHF) (Nelson-Aalen Estimator) represents the total accumulated risk up to time t. It is calculated by integrating the hazard function (h(t), the instantaneous risk of an event at time t, given survival up to t) from 0 to t.

ggsurvplot(fit2, conf.int = TRUE,

fun = "cumhaz", *# Key argument for plotting cumulative hazard*

risk.table = TRUE, risk.table.col = "strata")



Source: Figure by author(s).

**Figure 75**. Cumulative hazard function.

## More Advanced Aspects in Survival Analysis

### Competing Risk Analysis

In many clinical studies, patients are at risk for more than one type of event, and the occurrence of one event can prevent another from happening. This scenario is known as a competing risk setting. For example, patients with a pre-malignant condition may either progress to full-blown cancer or die from an unrelated cause before progression can occur. In this example, cancer progression the event of interest and death due to unrelated causes is a competing event.

The most common mistake in analyzing competing risks data is to treat the competing events as if they were standard censoring. This is incorrect because it violates the core assumption of non-informative censoring. When a patient who died of a heart attack is censored, that patient is no longer at risk for ever progressing to cancer. Treating this as censoring implicitly assumes the patient is still at risk, which leads to a biased and artificially inflated probability of the event of interest. The standard Kaplan-Meier method will overestimate the true incidence of the primary event in the presence of competing risks.

The below examples use the *mgus2* dataset. The question is: what factors (like age, sex, and hemoglobin levels) predict the cumulative incidence of cancer progression, properly accounting for death as a competing risk.

Now, we get the data ready for competing risk analysis:

*# Load the mgus2 dataset*

data(mgus2)

*# For simplicity, remove rows with missing hemoglobin*

mgus\_complete <- mgus2 %>% filter(!is.na(hgb))

*# The data has time to progression/death ('futime'),*

*# progression status ('pstat'), and death status ('dstat').*

*# We need to create a single competing risk status variable:*

*# 0 = censored (alive and no progression at last follow-up) # 1 = event of interest (progression to cancer)*

*# 2 = competing event (death without progression)*

mgus\_complete <- mgus\_complete %>%

mutate(

status\_cr = case\_when(

pstat == 1 ~ 1, *# Progression*

pstat == 0 & death == 1 ~ 2, *# Death without progression*

TRUE ~ 0 *# Otherwise, they are censored*

)

)

*# Check the event codes*

table(mgus\_complete$status\_cr)

##

## 0 1 2

## 402 114 855

#### Cumulative Incidence Function (CIF)

CIF would be a better approach for competing risk as the CIF for a specific event k is the probability that event k has occurred by time t, accounting for the fact that other competing events could have occurred instead. CIF may be calculated using *cuminc()* and CIF curves may be visualized using the *ggcompetingrisks()* function:

*# Calculate the CIF using the cuminc() function*

cif\_sex <- cuminc(ftime = mgus\_complete$futime,

fstatus = mgus\_complete$status\_cr,

group = mgus\_complete$sex,

cencode = 0)

*# Visualize the CIF curves*

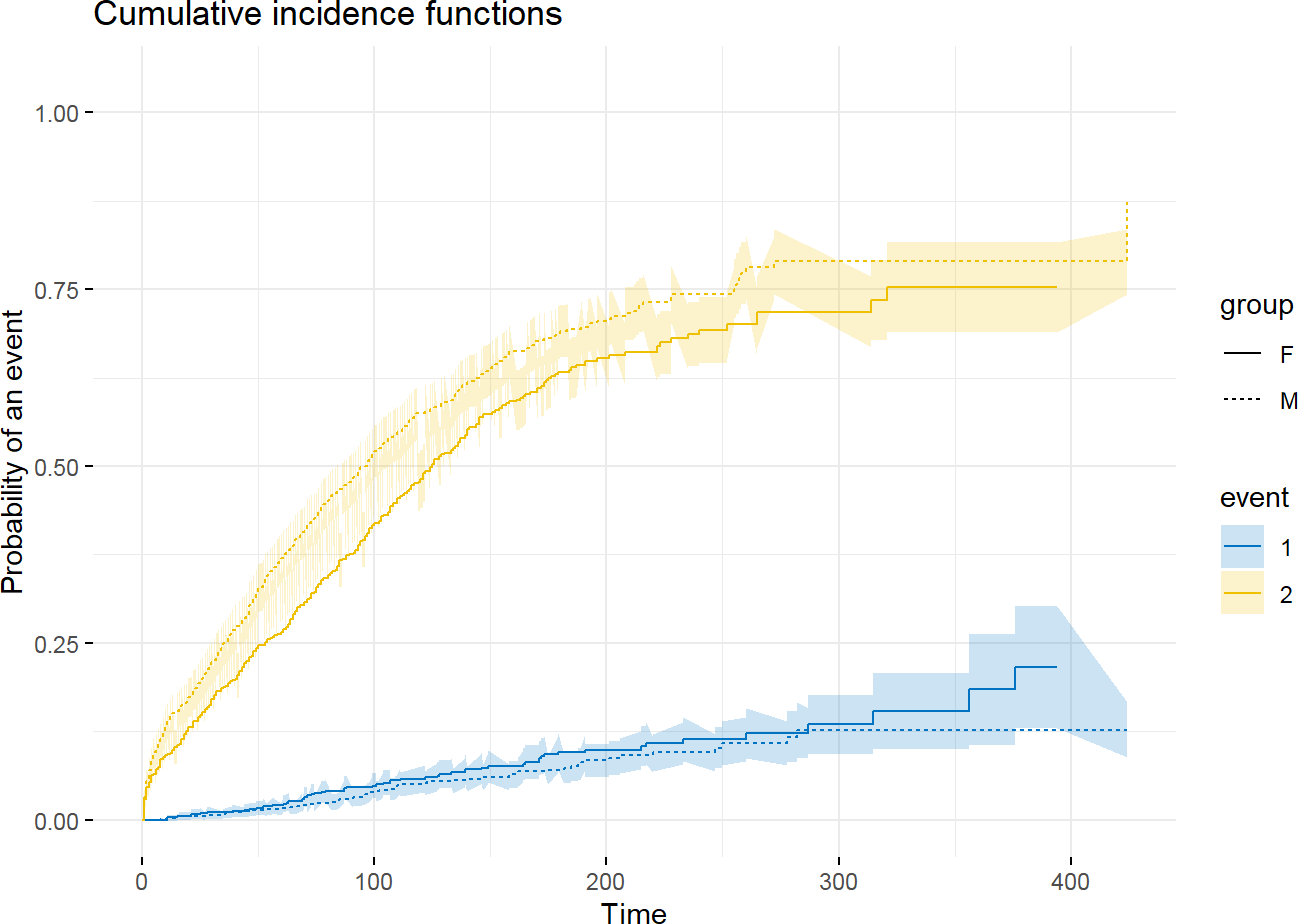
ggcompetingrisks(cif\_sex,

multiple\_panels = FALSE,

palette = "jco",

ggtheme = theme\_minimal(),

conf.int = TRUE)



Source: Figure by author(s).

**Figure 76**. Cumulative Incidence of Cancer Progression and Death by Sex.

The plot shows much higher incidence for death than for cancer progression highlighting the importance of the competing risk (death) in this elderly population.

#### Regression Modeling (Fine-Gray Model)

For assessing covariate effects, the Fine-Gray Subdistribution Hazard Model is the most common approach for clinical prediction. It directly models the effect of covariates on the cumulative incidence (CIF). The result is the Subdistribution Hazard Ratio (SHR). An SHR > 1 for a covariate means that an increase in that covariate is associated with a higher cumulative incidence of the event of interest [47], [49].

The example below uses Fine-Gray subdistribution hazards model to quantify the effect of “age”, “sex”, and hemoglobin (“hgb”) on the cumulative incidence of cancer progression accounting for death as a competing event, using the same *mgus* dataset. This may be done using the “*cmprsk”* package [47]:

*# Prepare the covariate matrix for the crr() function*

*# The function requires a matrix, not a formula*

cov\_matrix <- model.matrix(~ age + sex + hgb,

data = mgus\_complete)[,-1]

*# Fit the Fine-Gray subdistribution hazards model*

*# We specify failcode = 1 because progression is our event of interest*

fit\_fg <- crr(ftime = mgus\_complete$futime,

fstatus = mgus\_complete$status\_cr,

cov1 = cov\_matrix,

failcode = 1,

cencode = 0)

summary(fit\_fg)

## Competing Risks Regression ##

## Call:

## crr(ftime = mgus\_complete$futime, fstatus = mgus\_complete$status\_cr, ## cov1 = cov\_matrix, failcode = 1, cencode = 0)

##

## coef exp(coef) se(coef) z p-value

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | age -0.0174 | | 0.983 | 0.00579 -3.012 | 0.0026 |
| ## | sexM -0.2374 | | 0.789 | 0.18788 -1.264 | 0.2100 |
| ## | hgb -0.0110 | | 0.989 | 0.04481 -0.244 | 0.8100 |
| ## |  | |  |  |  |
| ## |  | exp(coef) | exp(-coef) 2.5% 97.5% | | |
| ## | age | 0.983 | 1.02 0.972 0.994 | | |
| ## | sexM | 0.789 | 1.27 0.546 1.140 | | |
| ## | hgb | 0.989 | 1.01 0.906 1.080 | | |
| ## |  |  |  | | |

## Num. cases = 1371

## Pseudo Log-likelihood = -767

## Pseudo likelihood ratio test = 7.01 on 3 df,

The output shows that SHR for age is 0.98 with p-value of 0.0026. This means that for every one-year increase in age at diagnosis, the subdistribution hazard of progressing to cancer *decreases* by about 0.02 %. This might seem counterintuitive, but it makes sense in a competing risks framework: older patients have a much higher risk of the competing event (death), leaving them less time and opportunity to progress to cancer.

### Time Dependent Covariates

Log-rank tests and Cox regression examine the association between survival outcomes and covariates of interest at baseline (before follow-up time and fixed at the start of the study).

Time dependent covariates (TDC) are variables whose values for a subject can change at different points during the observation period.

The example below uses the heart transplant *heart* data [50] of the survival package. Patients are enrolled in the study when they are accepted as a candidate for a transplant. The primary research question is: Does receiving a heart transplant improve survival for these patients? The key challenge is that “transplant status” is not a baseline characteristic. Every patient enters the study without a transplant. They are at risk of dying while waiting for a donor heart. At some point, a patient may receive a transplant. From that moment forward, their transplant status changes from “no” to “yes,” and they continue to be at risk of dying. A simple Cox model like *coxph(Surv(futime, fustat) ~ transplant)* would be flawed and lead to a spurious and overly optimistic estimate of the transplant’s benefit because it suffers from immortal time bias. It implicitly assumes that patients in the “transplant” group were somehow immortal from the start of the study until the moment they received their transplant.

Model for time dependent covariate may be done in the framework of the classical Cox proportional hazards model, but important adjustments to data is required: pre-processing the data into “start-stop” format. The *heart* dataset in R is conveniently pre-processed into this (*start, stop, event*) format.

fit\_tdc <- coxph(Surv(start, stop, event) ~ age + year + transplant,

data = heart)

summary(fit\_tdc)

## Call:

## coxph(formula = Surv(start, stop, event) ~ age + year + transplant, ## data = heart)

##

## n = 172, number of events = 75 ##

## coef exp(coef) se(coef) z Pr(>|z|)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | age | 0.02682 | 1.02718 | 0.01409 1.903 | | 0.0570 . |
| ## | year | -0.17870 | 0.83636 | 0.07044 -2.537 | | 0.0112 \* |
| ## | transplant1 | -0.03083 | 0.96964 | 0.31751 -0.097 | | 0.9226 |
| ## | --- |  |  |  | |  |
| ## ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 | | | | | |
| ## | exp(coef) exp(-coef) lower .95 upper .95 | | | | | |
| ## | age | 1.0272 | 0.9735 | 0.9992 | | 1.0559 |
| ## | year | 0.8364 | 1.1957 | 0.7285 | | 0.9602 |
| ## | transplant1 | 0.9696 | 1.0313 | 0.5204 | | 1.8066 |
| ## |  |  |  |  | |  |
| ## | Concordance = 0.618 (se = 0.034 ) | | | |  | |
| ## | Likelihood ratio test = 11.67 on 3 df, | | | | p = 0.009 | |
| ## | Wald test = 11.4 on 3 df, | | | | p = 0.01 | |
| ## | Score (logrank) test = 11.57 on 3 df, | | | | p = 0.009 | |

The output shows that at any given point in time, the instantaneous risk of death for a patient who has received a transplant is not significantly different from the risk for a patient who is still waiting. The above model assumes the effect of the transplant is constant over time which may not be true. One may assume that the benefit of a new heart might be different in the first month post-transplant compared to five years later. This may be tested by creating an interaction term between the transplant status and a function (e.g., logarithm) of time.

library(broom) *# for tidy output display*

fit\_tt <- coxph(Surv(start, stop, event) ~ age + year + transplant +

transplant \* log(stop),

data = heart)

*# Tidy summary with confidence intervals*

tidy\_fit <- tidy(fit\_tt, exponentiate = TRUE, conf.int = TRUE)

*# Rename & select useful columns*

tidy\_fit <- tidy\_fit %>%

transmute(

Term = term,

HR = estimate,

`95% CI Lower`

`95% CI Upper`

`Std. Error`

`z value` =

`p value` =

) %>%

=

= conf.low,

= conf.high, std.error,

statistic,

p.value

mutate(across(where(is.numeric), ~ round(.x, 4))) *# Round neatly*

*# Print as a nice table*

kable(tidy\_fit,

caption = 'Cox model with interaction term')%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 9**. Cox model with interaction term.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Term** | **HR** | **95% CI Lower** | **95% CI Upper** | **Std. Error** | **z Value** | **p Value** |
| age | 0.9881 | 0.9463 | 1.0317 | 0.0221 | −0.5444 | 0.5862 |
| year | 0.8434 | 0.6701 | 1.0616 | 0.1174 | −1.4508 | 0.1468 |
| transplant1 | 10.5581 | 0.3549 | 314.0688 | 1.7310 | 1.3616 | 0.1733 |
| log(stop) | 0.0000 | 0.0000 | 0.0000 | 44.7434 | −3.4412 | 0.0006 |
| transplant1:log(stop) | 0.7852 | 0.3356 | 1.8372 | 0.4337 | −0.5576 | 0.5771 |

Source: Table by author(s).

The output show that that the effect of “transplant” does not significantly change over logarithm of time.

## Useful Resources

* Tutorial for the essential R package which is the base of most packages related to survival analysis: [survival](https://cran.r-project.org/web/packages/survival/vignettes/survival.pdf) [43].
* Comprehensive book: [Applied Survival Analysis Using R](http://zums.ac.ir/files/socialfactors/files/Applied_Survival_Analysis_Using_R-2016.pdf) [48].
* “*[survminer](https://rpkgs.datanovia.com/survminer/index.html)*[”: Survival Analysis and Visualization](https://rpkgs.datanovia.com/survminer/index.html) package tutorial [44].

## Chapter summary

* “*survival*” (for most analysis & visualization), “*survminer*” (for publication ready visualization), “*lubridate*” (to work with date data) are the main R packages for survival analysis.
* *survfit(Surv(time, status))* to fit a Kaplan-Meier survival curve,

*ggsurvplot()* to visualize a survival curve, *survdiff()* to test difference between survival curves

* *coxph(Surv(time, status) ~...* for standard Cox model, *coxph(Surv(t1,*

*t2, stat)~...* for time dependent covariates.

* *cox.zph()* to check proportional hazards
* step() function for stepwise variable selection, cv.glmnet() function from the

glmnet package for performing LASSO penalized regression

* cmprsk is the main package for competing risk analysis with main function

crr() to fit the Fine-Gray subdistribution hazards model

* ggcompetingrisks() to visualize cumulative Incidence Function (CIF)

# Longitudinal Data Analysis with R

This chapter provides a practical guide to analyzing longitudinal data. It begins by exploring exploratory visualization techniques, such as spaghetti plots, to understand individual trajectories before introducing the two primary statistical frameworks for longitudinal analysis: Generalized Estimating Equations (GEE) and Generalized Linear Mixed-Effects Models (GLMM). The chapter distinguishes between the population-average focus of GEE models and the subject-specific focus of Mixed-Effects models, estimating fixed effects (population trends) and random effects (individual variations). The readers will gain the skills to implement these models using the key R packages “*geepack”* and “*lme4”*. Using a real-world dataset of Bangladeshi infants, the chapter walks through the entire workflow: from reshaping data and selecting appropriate correlation structures to fitting models and interpreting model outputs.

## Longitudinal Data

Longitudinal data are repeated measurements on the same individuals over time. Analyzing longitudinal data requires specialized statistical techniques because the repeated observations within an individual are typically correlated, violating the independence assumption of standard regression models. Failure to account for this within-subject correlation can lead to incorrect standard errors and misleading conclusions.

The main R packages for longitudinal data analysis introduced in this chapter are “*lme4*” for Linear Mixed Effect Models [37] and “*geepack*” for Generalized Estimation Equation models [51], and “*ggplot2*” for visualization [18].

We may also explore some supporting packages such as "*mgcv*" [52], "*gee*" [53], "*sjPlot*" [54], "*sjlabelled*" [55], "*sjmisc*" [56], "*MASS*" [57]. Some other supporting packages ("*rio*" [14], "*tidyverse*" [11], "*knitr*" [5], "*kableExtra*" [12]) are routinely used as mentioned in previous chapters. First, we load the required packages for longitudinal data analysis.

*#load multiple packages*

packages<-c("rio","tidyverse","knitr", "kableExtra", "lme4","mgcv",

"geepack","gee","ggplot2",

"sjPlot","sjlabelled","sjmisc","MASS")

lapply(packages, library, character.only = TRUE)

## Example: The Bangladesh Data

We will use the same dataset as in the "Practice Review" section of Chapter 2 “Introduction to R”. The dataset comes from a cohort study of 50 healthy Bangladeshi infants. This dataset is truly longitudinal, containing 996 stool samples and anthropometric data collected from these 50 infants over time.

Detailed data description may be found elsewhere [13].

We can load the example data from Github repository:

*# Define the URL for the Excel file*

url\_base <- "https://github.com/nhanhocu/metamicrobiomeR/raw/master/" data\_path <- "inst/extdata/QIIME\_outputs/Bangladesh/nature13421-s2.xlsx"

*# Import sheet 2 from the URL*

sh2 <- import(paste0(url\_base, data\_path), sheet = 2, skip = 1) str(sh2)

*#Rename column names* oldname<-colnames(sh2) newname<-c("cohort",

"family.id",

"child.id",

"sample.id",

"age.d",

"age.m", "whz","waz","haz", "breast.milk", "formula", "solid", "diarrhea", "antibiotic.7d", "medication", "no.sequence", "serun.id",

"barcode")

sh2<-sh2 %>% rename\_at(vars(oldname), ~newname)

*#colnames(sh2) #remove unused rows* sh2<-sh2 %>%

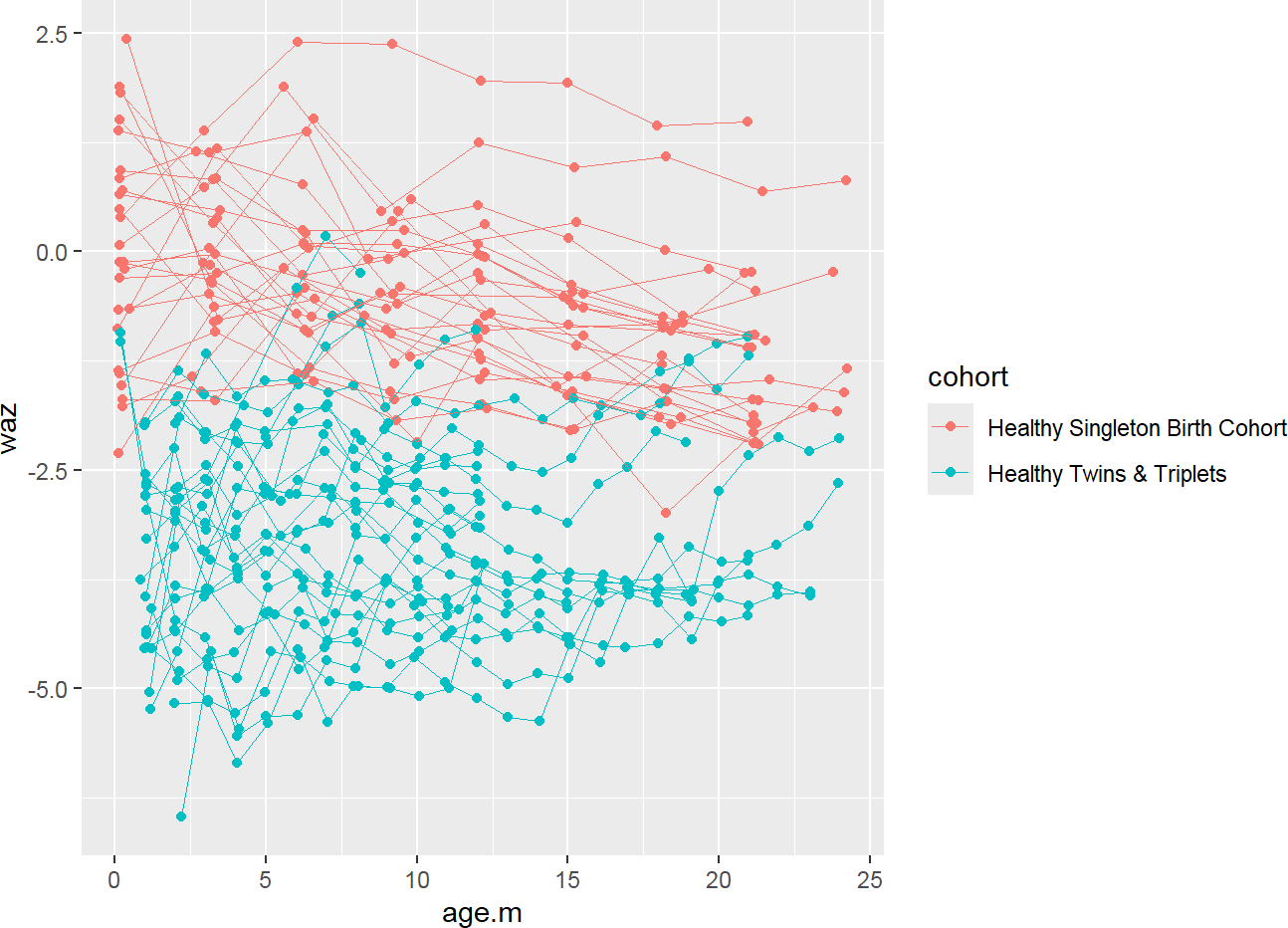
filter(!is.na(family.id))

### Exploratory Data Analysis

#### Longitudinal Data Visualization

Before modeling, we should visually examine the longitudinal data. The best way to start is with a "spaghetti plot," which draws a separate line for each individual subject. This helps us see both the overall trend and the individual-level trajectory.

Below is an example spaghetti plot for continuous variable weigh for age z-scores “waz” over time “age.m” using “*ggplot2”*. Note that *group = child.id* tells *ggplot()* to group the data by “child.id”, thus, *geom\_line()* knows to connect only the dots that belong to the same child, creating the characteristic "spaghetti" effect.



sh2 %>% filter(!is.na(waz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = waz, group = child.id, colour = cohort))+

geom\_line(aes(x = age.m, y = waz, group = child.id, colour = cohort),size = 0.3)

Source: Figure by author(s).

**Figure 77**. waz spaghetti plot.

The plot above shows the data's structure. We can see individual lines for each child, revealing that some children have consistently higher or lower “waz” scores than others. We also see the correlation: a child's “waz” at one time point is clearly related to their score at the next. The two cohort groups (Singletons vs. Twins/Triplets) appear to have very different trajectories, with the "Healthy Twins & Triplets" (blue) generally having lower “waz” scores than the "Healthy Singleton Birth Cohort" (red).

#### Plot the Means

To see the average population trend, we can plot the mean “waz” score at each month for each group.

The plot below shows, on average:

* + - * Trend: Weight for age z score (“waz”) decreases over time for singletons; not change in twins-triplets
      * Cross-sectional: singletons have larger “waz” than twins-triplets at every age
      * Longitudinal trend comparison: Decrease in average “waz” is larger for singletons then for twins & triplets.

d1<-sh2 %>% filter(!is.na(waz)) %>%

mutate(age.m.r = round(age.m,0)) %>% group\_by(age.m.r, cohort)%>% summarise(mean.waz = mean(waz,na.rm = T))

d1 %>%

ggplot()+

geom\_point(aes(x = age.m.r, y = mean.waz, colour = cohort))+ geom\_line(aes(x = age.m.r, y = mean.waz, colour = cohort),size = 0.3)



Source: Figure by author(s).

**Figure 78**. waz mean plot.

## Options for Analysis of Change

Our main research question is: Does mean change differ across groups?

### Pre-Post Analysis

One simple (but flawed) approach is to only use the first and last measurements for each child and see how much their “waz” changed.

* + - * Same baseline (e.g. randomization): analysis of post or change
      * Different baseline: analysis of post or change adjusting for baseline

This is generally not a good idea because it ignores all the valuable data from intermediate time points and fails to properly model the within-subject correlation.

### Linear Model for Time (the “naïve” approach)

Let's first try to answer our question using a standard linear regression *lm()* model, to demonstrate why it's problematic. We are treating all 573 valid data points as independent observations.

The Mean Model includes:

time = age or “age.m” in this data:

*E*[*Yij*|*ageij*] = *β*0 + *β*1*ageij*

The rate of Change is Slope *β*1 which is a constant rate of change.

Other models for time may be considered:

* + - * Quadratic
      * Linear spline
      * Cubic spline
      * Higher-order polynomials

Research Question

* + - * What is the rate of change?

Which is = time slope *E*[*wazij*|*xij* = (*age*.*m*, *cohort*)] = *β*0(*xij*) +

*β*1(*xij*)*timeij*

fitlm1<-lm(waz~age.m, data = sh2)

summary(fitlm1)

##

## Call:

## lm(formula = waz ~ age.m, data = sh2) ##

## Residuals:

## Min 1Q Median 3Q Max

## -4.134 -1.446 -0.114 1.333 4.758

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) -2.323415 0.134893 -17.224 <0.0000000000000002 \*\*\*

## age.m -0.005831 0.011508 -0.507 0.613

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 1.742 on 571 degrees of freedom

## (423 observations deleted due to missingness)

## Multiple R-squared: 0.0004494, Adjusted R-squared: -0.001301 ## F-statistic: 0.2567 on 1 and 571 DF, p-value: 0.6126

The output shows that “age.m” is not significant (p=0.613). This model is clearly not good as it's averaging the decreasing red trend and the stable blue trend of the two cohorts.

So, another question is:

* + - * Does the rate of change differ between singletons vs. twins-triplets? *E*[*Yij*] =

*β*0 + *β*1*ageij* + *β*2*cohorti* + *β*3*ageij* ∗ *cohorti*

fitlm2<-lm(waz~age.m +cohort +age.m\*cohort, data = sh2)

summary(fitlm2)

##

## Call:

## lm(formula = waz ~ age.m + cohort + age.m \* cohort, data = sh2) ##

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | Residuals: |  | | | | | | |
| ## | Min | 1Q | Median | 3Q | Max | | | |
| ## | -3.1604 -0.7730 | | -0.1336 | 0.7111 | 3.4986 |  |  |  |
| ## |  | |  |  |  |  |  |  |
| ## | Coefficients: | |  |  |  |  |  |  |
| ## |  | |  |  | Estimate | Std. Error | t | value |
| ## | (Intercept) |  | | | 0.11060 | 0.14276 | 0.775 | |
| ## | age.m |  | | | -0.05676 | 0.01110 | -5.113 | |
| ## | cohortHealthy | Twins & Triplets | | | -3.41607 | 0.17838 | -19.150 | |

|  |  |
| --- | --- |
| ## | age.m:cohortHealthy Twins & Triplets 0.05487 0.01469 3.734 |
| ## | Pr(>|t|) |
| ## | (Intercept) 0.438815 |
| ## | age.m 0.000000434 \*\*\* |
| ## | cohortHealthy Twins & Triplets < 0.0000000000000002 \*\*\* |
| ## | age.m:cohortHealthy Twins & Triplets 0.000207 \*\*\* |
| ## | --- |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 |
| ## | Residual standard error: 1.096 on 569 degrees of freedom |
| ## | (423 observations deleted due to missingness) |
| ## | Multiple R-squared: 0.6055, Adjusted R-squared: 0.6034 |
| ## | F-statistic: 291.1 on 3 and 569 DF, p-value: < 0.00000000000000022 |

Parameter Interpretation

* + - * *β*0 = 0.11060 = predicted “waz” for the reference group (Singletons) at age.m = 0.
      * *β*1 = -0.05676 = expected “waz” change (per month or 1 unit of “age.m”) for singletons (reference group)
      * *β*2 = -3.41607 = expected difference in “waz” comparing “Healthy Twins & Triplets” to singletons at first visit (newborn)
      * *β*3 = 0.05487 = expected difference in “waz” change (per month) between “Healthy Twins & Triplets” and singletons
      * Calculated Slope for “Healthy Twins & Triplets”: Slope(Twins) = Slope(Singletons) + Slope Difference = -0.05676 + 0.05487 = -0.0019
      * The interaction is highly significant (p=0.0002), confirming this difference in slopes between the two cohorts is real.

Linear regression line may be added to the plot with *stat\_smooth(…, method = “glm”)*:

sh2 %>% filter(!is.na(waz)) %>% ggplot()+

geom\_point(aes(x = age.m, y = waz, group = child.id, colour = cohort))+

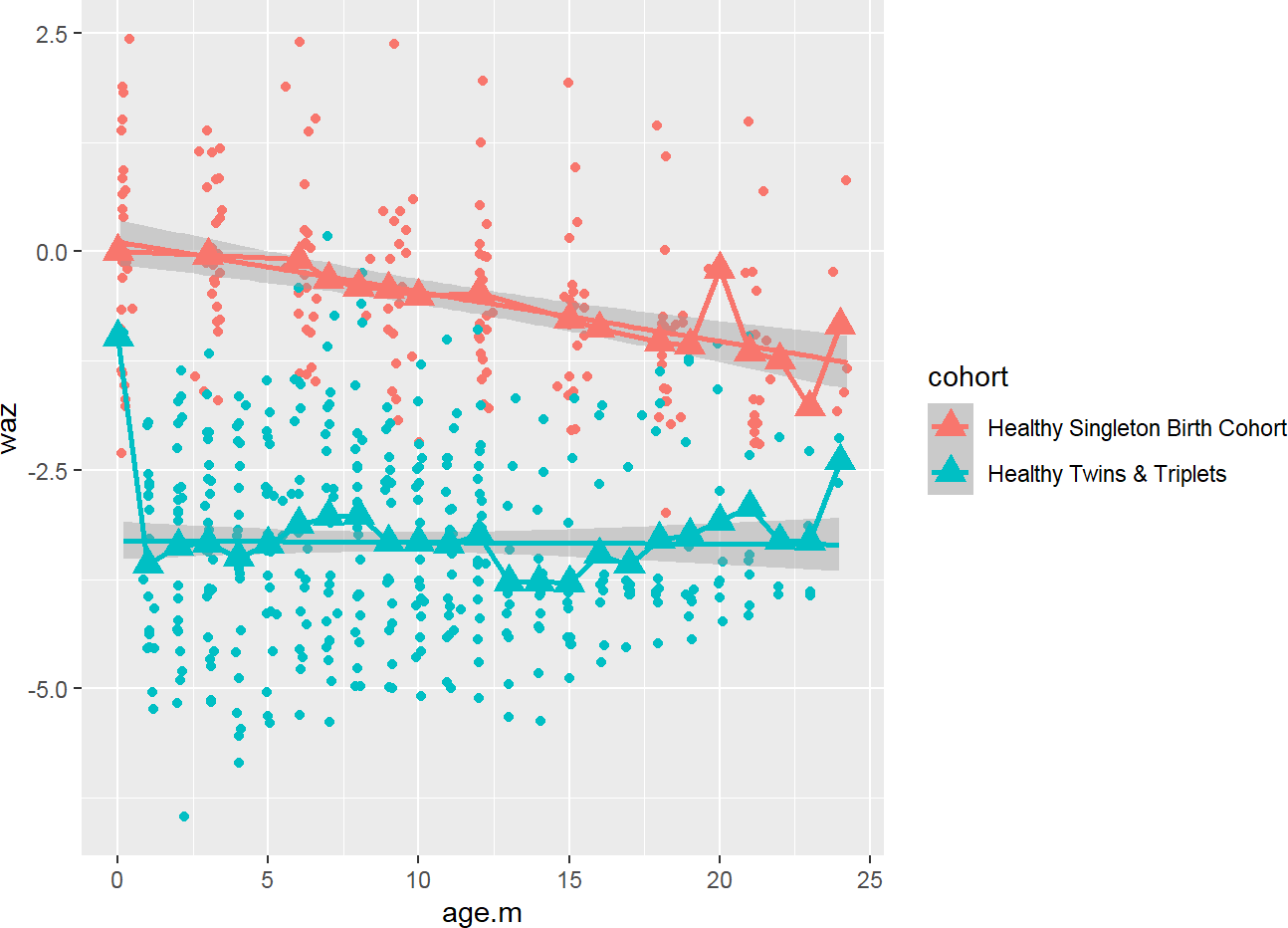
stat\_smooth(aes(x = age.m, y = waz,

colour = cohort), method = 'glm',size = 1)+ geom\_point(aes(x = age.m.r, y = mean.waz,

colour = cohort), size = 4, pch = 17, data = d1)+

geom\_line(aes(x = age.m.r, y = mean.waz,

colour = cohort), size = 1, pch = 17, data = d1)



Source: Figure by author(s).

**Figure 79**. waz linear regression plot.

### Issues of Linear Model

The below are not accounted in linear model:

Dependence & Correlation

* + - * Response variables measured on the same subject are correlated.
      * Observations are dependent or correlated when one variable does predict the value of another variable (“waz” at month 1 may predict “waz” at month 2)
      * The correlation is a measure of dependence that takes values between −1 and

+1

A correlation of 0 implies that two measures are unrelated (linearly)

* + - * A correlation of 1 implies that the two measures fall perfectly on a line (one exactly predicts the other!)

Variance

The variance measures the average distance that an observation falls away from the mean.

Covariance

The covariance measures whether, on average, departures in one variable ‘go together with’ departures in a second variable.

With the above issues of the linear regression model, we need a method that correctly handles longitudinal data.

## Generalized Estimating Equations (GEE)

### Introduction to GEE

The GEE model contrasts average outcome values across populations of individuals defined by covariate values, while accounting for correlation. GEE is an extension of generalized linear models (GLMs) and is particularly useful when the primary interest is in the effect of covariates on the average response in the population, rather than on individual-level trajectories [58].

The assumptions of GEE are:

* + - * Observations are independent across subjects
      * Observations may be correlated within subjects: longitudinal correlation

structure is a nuisance feature of the data. The GEE model is specified by:

* + - * A mean model: A regression model for the average outcome, e.g., linear, logistic;
      * A correlation model: A model for longitudinal correlation, which describes how the repeated measurements within a subject are correlated. Common choices for the working correlation structure include:
* Independence: correlation is assumed to be zero (observations over time are independent). This is appropriate with use of robust variance estimator (large n).
* Exchangeable: correlation is assumed to be constant (all observations over time have the same correlation). This is more appropriate for clustered data.
* Auto-regressive: correlation is assumed to depend on time or distance (correlation decreases as a power of how many timepoints apart two observations are). This is more appropriate for equally-spaced longitudinal data.
* Unstructured: correlation is assumed to be distinct for each pair (correlation between all timepoints may be different). This is only appropriate for short series (small m) on many subjects (large n).

An advantage of GEE is that the estimates of the regression coefficients are generally consistent even if the working correlation structure is mis-specified, although the efficiency of the estimates is improved with a correctly specified correlation structure.

### GEE Example: The Bangladesh Data

Objectives

In this example, we characterize “waz” (continuous outcome) change among singletons and twins-triplets:

* + - * Estimate the average “waz” change among all children
      * Estimate the “waz” change for individual children
      * Characterize the degree of heterogeneity across children
      * Identify factors that predict “waz” change

#### GEE Approach

* + - * Examine “waz” change among twins\_triplets and singletons: *E*[*Yij*] = *β*0 +

*β*1*ageij* + *β*2*cohorti* + *β*3*ageij* ∗ *cohorti*

* + - * Consider various specifications for the ‘working’ correlation structure. Selection

of a working correlation structure should be guided by a prior knowledge and/or exploratory analysis. Different Correlation Structures may be tried with the GEE models. In the examples below, we refit the GEE models for different specified correlation structures. We also refit the linear model for comparison.

We will use the *geeglm()* function from the “*geepack*” package [51] to fit GEE models.

* *id*: id of subjects (which observations belong to the same cluster)
* *family = gaussian*: for continuous outcome (*family = binomial* for binary outcome, *family = poisson* with *offset = log(time)* for count outcome)
* *corstr*: for correlation structure

*#library(geepack)*

waz.nona<-sh2 %>%

filter(!is.na(waz))

g.i<-geeglm(waz~age.m +cohort +age.m\*cohort, id = child.id,

family = gaussian,

corstr = "independence",

data = waz.nona)

summary(g.i)

##

## Call:

## geeglm(formula = waz ~ age.m + cohort + age.m \* cohort, family = gaussian, ## data = waz.nona, id = child.id, corstr = "independence")

##

## Coefficients:

## Estimate

## (Intercept) 0.110603312666482406224

## age.m -0.056757575247523540940

## cohortHealthy Twins & Triplets -3.416067033639933470113 ## age.m:cohortHealthy Twins & Triplets 0.054872161140891988207 ## Std.err

## (Intercept) 0.000000000000000446312

## age.m 0.000000000000000041738

## cohortHealthy Twins & Triplets 0.000000000000000071888 ## age.m:cohortHealthy Twins & Triplets 0.000000000000000003328

## Wald

## (Intercept) 61412838741582766818648662002

## age.m 1849199462013278828642420806208

## cohortHealthy Twins & Triplets 2258111572585936842822006004640424 ## age.m:cohortHealthy Twins & Triplets 271818077730298728628640860480882

## Pr(>|W|)

## (Intercept) <0.0000000000000002 \*\*\*

## age.m <0.0000000000000002 \*\*\*

## cohortHealthy Twins & Triplets <0.0000000000000002 \*\*\* ## age.m:cohortHealthy Twins & Triplets <0.0000000000000002 \*\*\* ## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 ##

## Correlation structure = independence ## Estimated Scale Parameters:

##

## Estimate Std.err

## (Intercept) 1.193 0.00000000000000003975

## Number of clusters: 1 Maximum cluster size: 573

g.e<-geeglm(waz~age.m +cohort +age.m\*cohort,

id = child.id,

family = gaussian,

corstr = "exchangeable",

data = waz.nona)

summary(g.e)

##

## Call:

## geeglm(formula = waz ~ age.m + cohort + age.m \* cohort, family = gaussian, ## data = waz.nona, id = child.id, corstr = "exchangeable")

##

## Coefficients:

## Estimate Std.err Wald

## (Intercept) 0.1106 NaN NaN

## age.m -0.0568 NaN NaN

## cohortHealthy Twins & Triplets -3.4161 NaN NaN ## age.m:cohortHealthy Twins & Triplets 0.0549 NaN NaN ## Pr(>|W|)

## (Intercept) NaN

## age.m NaN

## cohortHealthy Twins & Triplets NaN ## age.m:cohortHealthy Twins & Triplets NaN ##

## Correlation structure = exchangeable ## Estimated Scale Parameters:

##

## Estimate Std.err ## (Intercept) 1.19 0.000000000000000104

## Link = identity ##

## Estimated Correlation Parameters:

## Estimate Std.err ## alpha -0.00175 0.000000000000000000339

## Number of clusters: 1 Maximum cluster size: 573

g.a<-geeglm(waz~age.m +cohort +age.m\*cohort,

id = child.id,

family = gaussian, corstr = "ar1",

data = waz.nona) summary(g.a)

##

## Call:

## geeglm(formula = waz ~ age.m + cohort + age.m \* cohort, family = gaussian, ## data = waz.nona, id = child.id, corstr = "ar1")

##

## Coefficients:

## Estimate

## (Intercept) -0.04991613939

## age.m -0.05447299583

## cohortHealthy Twins & Triplets -3.50917501940 ## age.m:cohortHealthy Twins & Triplets 0.08526818243

## Std.err

## (Intercept) 0.00000196221

## age.m 0.00000000123

## cohortHealthy Twins & Triplets 0.00000237983 ## age.m:cohortHealthy Twins & Triplets 0.00000004484 ## Wald

## (Intercept) 647128510

## age.m 1957587943802740

## cohortHealthy Twins & Triplets 2174293484283 ## age.m:cohortHealthy Twins & Triplets 3615941311647

## Pr(>|W|)

## (Intercept) <0.0000000000000002 \*\*\*

## age.m <0.0000000000000002 \*\*\*

## cohortHealthy Twins & Triplets <0.0000000000000002 \*\*\* ## age.m:cohortHealthy Twins & Triplets <0.0000000000000002 \*\*\* ## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 ##

## Correlation structure = ar1 ## Estimated Scale Parameters:

##

## Estimate Std.err ## (Intercept) 1.23 0.000000308

## Link = identity ##

## Estimated Correlation Parameters:

## Estimate Std.err ## alpha 0.827 0.000000213

## Number of clusters: 1 Maximum cluster size: 573

*# Linear model*

*# = > SE are larger*

fit.ols<-lm(waz~age.m +cohort +age.m\*cohort,

data = waz.nona)

summary(fit.ols)

##

## Call:

## lm(formula = waz ~ age.m + cohort + age.m \* cohort, data = waz.nona) ##

## Residuals:

## Min 1Q Median 3Q Max ## -3.160 -0.773 -0.134 0.711 3.499 ##

## Coefficients:

## Estimate Std. Error t value

## (Intercept) 0.1106 0.1428 0.77

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | age.m | -0.0568 | 0.0111 | -5.11 |
| ## | cohortHealthy Twins & Triplets | -3.4161 | 0.1784 | -19.15 |
| ## | age.m:cohortHealthy Twins & Triplets | 0.0549 | 0.0147 | 3.73 |
| ## |  |  | Pr(>|t|) |  |
| ## | (Intercept) |  | 0.43882 |  |
| ## | age.m 0.00000043 \*\*\* | | | |
| ## | cohortHealthy Twins & Triplets < 0.0000000000000002 \*\*\* | | | |
| ## | age.m:cohortHealthy Twins & Triplets 0.00021 \*\*\* | | | |
| ## | --- | | | |
| ##  ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 | | | |
| ## | Residual standard error: 1.1 on 569 degrees of freedom | | | |
| ## | Multiple R-squared: 0.605, Adjusted R-squared: 0.603 | | | |
| ## | F-statistic: 291 on 3 and 569 DF, p-value: <0.0000000000000002 | | | |

The outputs show that the standard errors (SE) of GEE models are smaller than those of linear models. The interpretation of the coefficients from GEE models would be similar to that from linear model (see Chapter: Linear regression with R, or see the example using linear regression model above).

The correlation structure is selected based on the smallest Quasi Information Criterion (QIC) using *QIC()*.

QIC(g.i)

## QIC

## 683.5336302609380254580173641443253

## QICu

## 691.5336302609380254580173641443253

## Quasi Lik

## -341.7668151304690127290086820721626

## CIC

## 0.0000000000000000000000000000303

## params

## 4.0000000000000000000000000000000

## QICC

## 684.4225191498269396106479689478874

QIC(g.e)

|  |  |  |
| --- | --- | --- |
| ## | QIC | QICu |
| ## | 683.5336302609380255 | 691.5336302609380255 |
| ## | Quasi Lik | CIC |
| ## | -341.7668151304690127 | -0.0000000000000125 |
| ## | params | QICC |
| ## | 4.0000000000000000 | 684.8972666245743994 |

QIC(g.a) ## QIC QICu Quasi Lik

## 702.140678184319 710.140678182590 -351.070339091295

## CIC params QICC ## 0.000000000865 4.000000000000 703.504314547955

### GEE Example: Other Types of Outcomes

#### Binary Outcome

The example below uses the *respiratory* dataset from the “*geepack”* package [51]. Our research question is: On average, what is the effect of an active treatment A (vs. placebo P) on the likelihood of a good respiratory outcome (1 = good, 0 = poor), accounting for age and visit number? Data is clustered by patient id (*id=id*). We tell *geeglm()* to perform a longitudinal logistic regression with *family = binomial*.

*#library(geepack)*

*# Load the built-in respiratory dataset*

data("respiratory")

str(respiratory)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | 'data.frame': | | 444 obs. | | | of 8 | variables: |
| ## | $ | center : | int | 1 | 1 1 1 | 1 1 1 | 1 1 1 ... |
| ## | $ | id : | int | 1 | 1 1 1 | 2 2 2 | 2 3 3 ... |

## $ treat : Factor w/ 2 levels "A","P": 2 2 2 2 2 2 2 2 1 1 ...

## $ sex : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 2 2 2 ...

## $ age : int 46 46 46 46 28 28 28 28 23 23 ...

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | $ | baseline: | int | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | ... |
| ## | $ | visit : | int | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | ... |
| ## | $ | outcome : | int | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | ... |

respiratory$treat<-factor(respiratory$treat, levels = c("P","A"))

*# Fit the GEE model with an exchangeable correlation structure # Exchangeable assumes the correlation between*

*# any two visits is the same for a child.*

gee\_binomial <- geeglm(

outcome ~ age + treat + visit,

id = id,

data = respiratory,

family = binomial,

corstr = "exchangeable"

)

*# Display the summary*

summary(gee\_binomial)

##

## Call:

## geeglm(formula = outcome ~ age + treat + visit, family = binomial, ## data = respiratory, id = id, corstr = "exchangeable")

##

## Coefficients:

## Estimate Std.err Wald Pr(>|W|)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | (Intercept) | 0.3328 | 0.4630 0.52 | 0.4722 |
| ## | age | -0.0121 | 0.0116 1.08 | 0.2986 |
| ## | treatA | 0.9804 | 0.3128 9.83 | 0.0017 \*\* |
| ## | visit | -0.0625 | 0.0656 0.91 | 0.3412 |
| ## | --- |  |  |  |

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 ##

## Correlation structure = exchangeable ## Estimated Scale Parameters:

##

## Estimate Std.err ## (Intercept) 1 0.0548 ## Link = identity

##

## Estimated Correlation Parameters:

## Estimate Std.err ## alpha 0.491 0.0653

## Number of clusters: 111 Maximum cluster size: 4

The coefficients are in log-odds scale. We must exponentiate them to get odds ratios.

exp(coef(gee\_binomial))

|  |  |  |  |
| --- | --- | --- | --- |
| ## (Intercept) | age | treatA | visit |
| ## 1.395 | 0.988 | 2.665 | 0.939 |

The output shows that the odds ratio is 2.665, or the odds of having a good respiratory outcome for a child receiving the active treatment (A) are 2.665 the odds of a child receiving the placebo.

#### Count Outcome

The below example uses the “*epil*” dataset from the “*MASS”* package [57]. Our research question is: On average, does a new drug (“treat” = “progabide”) reduce the rate of seizures compared to a placebo, after accounting for baseline seizure count (“base”)? We specify a model for count data by *family = poisson*, and a crucial step for modeling rates is *offset = log(period)*. By including the log of the observation period as an offset, the model is no longer predicting the raw count (*y*) but the rate (*y / period*).

*#library(geepack)*

*#library(MASS) # For the epil dataset*

*# Load and inspect the data*

data("epil") str(epil)

## 'data.frame': 236 obs. of 9 variables: ## $ y : num 5 3 3 3 3 5 3 3 2 4 ...

## $ trt : Factor w/ 2 levels "placebo","progabide": 1 1 1 1 1 1 1 1 1 1 ...

## $ base : int 11 11 11 11 11 11 11 11 6 6 ...

## $ age : int 31 31 31 31 30 30 30 30 25 25 ...

## $ V4 : int 0 0 0 1 0 0 0 1 0 0 ...

## $ subject: int 1 1 1 1 2 2 2 2 3 3 ...

## $ period : int 1 2 3 4 1 2 3 4 1 2 ...

## $ lbase : num -0.756 -0.756 -0.756 -0.756 -0.756 ...

## $ lage : num 0.1142 0.1142 0.1142 0.1142 0.0814 ...

|  |  |  |
| --- | --- | --- |
| *# Fit the Poisson GEE model*  *# Note the use of offset() to model the rate of seizures, # not just the count.*  gee\_poisson <- geeglm( y ~ base + trt,  id = subject,  data = epil, | | |
| family | = | poisson, |
| corstr | = | "ar1", |
| offset | = | log(period) *# Model rate: seizures per week* |
| ) |  |  |
| *# Display the summary*  summary(gee\_poisson) | | |

##

## Call:

## geeglm(formula = y ~ base + trt, family = poisson, data = epil, ## offset = log(period), id = subject, corstr = "ar1")

##

## Coefficients:

## Estimate Std.err Wald Pr(>|W|) ## (Intercept) 0.30161 0.12681 5.66 0.017 \*

## base 0.02180 0.00104 442.88 <0.0000000000000002 \*\*\*

## trtprogabide -0.23657 0.17040 1.93 0.165

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 ##

## Correlation structure = ar1 ## Estimated Scale Parameters:

##

## Estimate Std.err ## (Intercept) 10.5 2.1

## Link = identity ##

## Estimated Correlation Parameters:

## Estimate Std.err ## alpha 0.353 0.0728

## Number of clusters: 59 Maximum cluster size: 4

The coefficients are in log scale. We must exponentiate them to get interpretable Incident Rate Ratios (IRRs).

## (Intercept) base trtprogabide

*# Exponentiate to get Incident Rate Ratios (IRR)*

exp(coef(gee\_poisson))

## 1.352 1.022 0.789

The output shows that the IRR is 0.789 or, on average, the seizure rate for patients in the progabide group is 0.789 of the seizure rate for patients in the placebo group.

### GEE Summary Main Characteristics

* + - * Primary focus of the analysis is a marginal mean regression model that corresponds to any GLM
      * Longitudinal correlation is secondary and is treated as a nuisance feature
      * Need ‘working’ correlation model
      * Semi-parametric: Only the mean and correlation models are specified
      * Hypothesis testing with GEE uses Wald statistics
      * Working correlation model does not need to be correctly specified to obtain a

consistent estimator for *β* or valid standard errors for *β*ˆ, but efficiency gains are possible if the correlation model is correct.

Issues

* + - * Only one source of correlation is addressed: longitudinal or cluster
      * Any missing data are required to be missing completely at random
      * Issues arise with time-dependent exposures and covariance weighting

## Generalized Linear Mixed-Effects Models (GLMM)

### GLMM Overview

Linear mixed-effects models (LME) contrast outcomes both within and between individuals [59].

LME models include the following components:

* + - * Fixed Effects: These are the conventional regression coefficients and are assumed to be constant, or “fixed,” across the entire population. They represent the population-average relationship between a predictor and the outcome. For the Bangladesh data example, we estimate the average “waz” change by a unit increase in age.
      * Random Effects: These are effects that are specific to each individual or cluster

(e.g., each subject). Random effects capture the heterogeneity across subjects, acknowledging that each individual may have a unique trajectory.

LME models decompose the total variance in the outcome into several components:

* + - * Between-Subject Variance: This is the variability captured by the random effects. It explains why some subjects consistently have higher or lower values than others (random intercept) or why some subjects respond more strongly to a predictor than others (random slope).
      * Within-Subject Variance (Residual Variance): This is the variability of a

subject’s measurements around their own individual trajectory. It represents measurement error and other unexplained time-specific fluctuations.

Generalized Linear Mixed-Effects Models (GLMM) is a powerful extension of the Linear Mixed-Effects (LME) model. GLMM handles non-normal outcome variables (e.g., binary, count) by using a link function to transform the outcome and an appropriate error distribution (called a “family”).

### Linear Mixed Effect (LME) Model Example

#### The Bangladesh Data

We go back to our previous data example to characterize “waz” change among singletons and twins-triplets:

* + - * Estimate the average “waz” change among all children
      * Estimate the “waz” change for individual children
      * Characterize the degree of heterogeneity across children
      * Identify factors that predict “waz” change

The main R package for LME model is “*lme4”*. We use the *lmer()* function from the “*lme4”* package for continuous outcomes (LME is a specific type of GLMM) [37].

#### Linear mixed effect (LME) model with random intercept

This model assumes a common slope for everyone in a cohort but allows each child to have their own unique baseline (intercept). The random effect formula is specified by *(1 | child.id)*. It reads: "fit a random intercept (the 1) for each level of *child.id*."

*#library(lme4)*

*#library(lmerTest) #for p-value*

waz.nona<-sh2 %>%

filter(!is.na(waz))

r.in<-lmer(waz ~ age.m\*cohort+ (1 | child.id), data = waz.nona)

summary(r.in)

## Linear mixed model fit by REML ['lmerMod']

## Formula: waz ~ age.m \* cohort + (1 | child.id)

## Data: waz.nona

##

## REML criterion at convergence: 1344

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -4.437 -0.565 -0.025 0.552 4.973

##

## Random effects:

## Groups Name Variance Std.Dev.

## child.id (Intercept) 0.721 0.849

## Residual 0.464 0.681

## Number of obs: 573, groups: child.id, 50

##

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ##  ## | Fixed effects: | Estimate | Std. Error | t | value |
| ## | (Intercept) | 0.09962 | 0.19176 |  | 0.52 |
| ## | age.m | -0.05498 | 0.00697 |  | -7.89 |
| ## | cohortHealthy Twins & Triplets | -3.47286 | 0.26529 |  | -13.09 |
| ## | age.m:cohortHealthy Twins & Triplets 0.06998 | | 0.00969 | 7.23 | |
| ## |  | |  |  | |
| ## | Correlation of Fixed Effects: | |  |  | |
| ## | (Intr) age.m chHT&T | |  |  | |
| ## | age.m -0.386 | |  |  | |
| ## | chrtHlthT&T -0.723 0.279 | |  |  | |
| ## | ag.m:chHT&T 0.277 -0.719 -0.354 | |  |  | |

Note that, the package “*lmerTest”* should be loaded to get p-values for fixed effects.

The output shows random effects and fixed effects:

Random effects: *child.id (Intercept)*: The Variance is 0.721. This is the between-subject variance. It quantifies the variability in the intercepts from child to child. *Residual*: The Variance is 0.464. This is the within-subject variance, or the unexplained "noise" of measurements around each child's own line.

Fixed effects: The coefficients (Intercept), “age.m”, etc., are the population-average estimates. They are interpreted just like the *lm()* and GEE models:

* + - * Intercept coefficient: which means that population-average “waz” of the reference group (singleton) is 0.1 when age.m is zero (the start of the study).
      * “age.m” coefficient: which means that the estimated slope (rate of waz change) for one unit increase in age.m of the reference group is −0.055.
      * “cohortHealthy Twins & Triplets” coefficient: which means that the estimated

difference in waz between the Healthy Twins & Triplets group and the reference (singleton) group at age.m = 0 is −3.473.

* + - * “age.m:cohortHealthy Twins & Triplets” coefficient: which means that the

difference in slopes between the Healthy Twins & Triplets group and the reference group is 0.07.

The slope for Healthy Twins & Triplets group may be calculated by = −0.055 +

0.07 = 0.015

#### LME model with random intercept and random slope

This model is more complex. It allows each child to have both their own baseline (intercept) and their own unique rate of change (slope). We specify in the model formula *(age.m | child.id)*: This is shorthand for *(1 + age.m | child.id)*. It reads: "fit a random intercept and a random slope for “age.m”, grouping by “child.id”."

r.s<-lmer(waz ~ age.m\*cohort+ (age.m | child.id), data = waz.nona)

summary(r.s)

## Linear mixed model fit by REML ['lmerMod']

## Formula: waz ~ age.m \* cohort + (age.m | child.id)

## Data: waz.nona

##

## REML criterion at convergence: 1239

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -2.998 -0.510 0.007 0.525 4.531

##

## Random effects:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | Groups Name Variance Std.Dev. Corr |  | | |
| ## | child.id (Intercept) 0.9505 0.975 |
| ## | age.m 0.0126 0.112 -0.44 |
| ## | Residual 0.2939 0.542 |
| ## | Number of obs: 573, groups: child.id, 50 |
| ## |  |
| ## | Fixed effects: |
| ## | Estimate | Std. Error | t | value |
| ## | (Intercept) 0.1027 | 0.2077 |  | 0.49 |
| ## | age.m -0.0549 | 0.0232 |  | -2.37 |
| ## | cohortHealthy Twins & Triplets -3.6032 | 0.2913 |  | -12.37 |
| ## | age.m:cohortHealthy Twins & Triplets 0.0997 | 0.0332 |  | 3.01 |
| ## |  |  |  |  |
| ## | Correlation of Fixed Effects: |  |  |  |

## (Intr) age.m chHT&T

## age.m -0.475

## chrtHlthT&T -0.713 0.339

## ag.m:chHT&T 0.332 -0.699 -0.479

## optimizer (nloptwrap) convergence code: 0 (OK)

## Model failed to converge with max|grad| = 0.00410411 (tol = 0.002, component 1)

The output shows random effects and fixed effects:

Random effects: *child.id (Intercept)*: Variance of the intercepts = 0.9505. *age.m*: Variance of the “age.m” slopes = 0.0126. This tells us that the slopes for “age.m” vary from child to child. There is a “moderate” correlation (Corr: -0.44) between random slopes and intercepts. This means that a subject’s baseline “waz” is moderately correlated with the change of “waz” over age (children who started with a higher-than-average “waz” (positive intercept) tended to have a more-negative-than-average slope (a faster decline)). If a strong correlation is observed, it is “over-parameterized” (too many parameters). In such cases, either intercept or slope is implemented as a random effect.

Fixed effects: are interpreted as above.

We can also designate that intercepts and slopes are determined independently.

r.isi<-lmer(waz ~ age.m\*cohort+ (1 | child.id)+

(0+age.m | child.id),

data = waz.nona) summary(r.isi)

## Linear mixed model fit by REML ['lmerMod']

## Formula:

## waz ~ age.m \* cohort + (1 | child.id) + (0 + age.m | child.id)

## Data: waz.nona

##

## REML criterion at convergence: 1248

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -3.369 -0.515 0.002 0.532 4.608

##

## Random effects:

## Groups Name Variance Std.Dev.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | child.id (Intercept) 0.8735 0.935 |  | | |
| ## | child.id.1 age.m 0.0105 0.102 |
| ## | Residual 0.2990 0.547 |
| ## | Number of obs: 573, groups: child.id, 50 |
| ## |  |
| ## | Fixed effects: |
| ## | Estimate | Std. Error | t | value |
| ## | (Intercept) 0.1019 | 0.2003 |  | 0.51 |
| ## | age.m -0.0548 | 0.0213 |  | -2.58 |
| ## | cohortHealthy Twins & Triplets -3.5907 | 0.2807 |  | -12.79 |
| ## | age.m:cohortHealthy Twins & Triplets 0.0972 | 0.0305 |  | 3.19 |
| ## |  |  |  |  |
| ## | Correlation of Fixed Effects: |  |  |  |

## (Intr) age.m chHT&T

## age.m -0.081

## chrtHlthT&T -0.714 0.058

## ag.m:chHT&T 0.057 -0.698 -0.086

Comparing the model with random intercept vs. the model with random intercept and slope: Is the more complex random slope model (*r.s*) significantly better than the simpler random intercept model (*r.in*)? We use a Likelihood Ratio Test (*anova()*).

anova(r.in,r.s)

|  |  |
| --- | --- |
| ## | Data: waz.nona |
| ## | Models: |
| ## | r.in: waz ~ age.m \* cohort + (1 | child.id) |
| ## | r.s: waz ~ age.m \* cohort + (age.m | child.id) |
| ## | npar AIC BIC logLik -2\*log(L) Chisq Df |
| ## | r.in 6 1337 1363 -662 1325 |
| ## | r.s 8 1240 1275 -612 1224 100 2 |
| ## | Pr(>Chisq) |
| ## | r.in |
| ## | r.s <0.0000000000000002 \*\*\* |
| ## | --- |
| ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1 |

The output shows that random slope is significant p-value (<0.000...) meaning that the model with random intercept and random slope (*r.s*) better fit the data than the *r.in* model (with only a random intercept). This indicates that children not only start at different baseline, but their “waz” also changes at different rates.

#### Extract effects

We can get population-average (fixed) coefficients and subject-specific (random) effects:

*#fixed effects*

fixef(r.s)

## (Intercept)

## 0.1027

## age.m

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## |  | |  | | -0.0549 | | | |
| ## | cohortHealthy | | | Twins | | & | Triplets | |
| ## |  | |  | | -3.6032 | | | |
| ## | age.m:cohortHealthy | | | Twins | | & | Triplets | |
|  |  | | |  | |  |  | |
| ## |  | |  | |  | | | 0.0997 |
| *#random effects* | | |  | |  | | |  |
| ranef(r.s) | | |  | |  | | |  |
| ## | $child.id | |  | |  | | |  |
| ## |  | (Intercept) | | | age.m | | | |
| ## | Bgsng7018 | 0.1813 | | | -0.01865 | | |  |
| ## | Bgsng7035 | -1.4131 | | | 0.02076 | | |  |
| ## | Bgsng7052 | -0.1744 | | | 0.02352 | | |  |
| ## | Bgsng7063 | 0.8686 | | | -0.04558 | | |  |
| ## | Bgsng7071 | 1.0960 | | | -0.01241 | | |  |
| ## | Bgsng7082 | -0.7357 | | | 0.04352 | | |  |
| ## | Bgsng7090 | 0.2326 | | | -0.00878 | | |  |
| ## | Bgsng7096 | -0.3517 | | | -0.02508 | | |  |
| ## | Bgsng7106 | 0.0299 | | | -0.02547 | | |  |
| ## | Bgsng7114 | -0.5781 | | | 0.03357 | | |  |
| ## | Bgsng7115 | 0.5440 | | | -0.01088 | | |  |
| ## | Bgsng7128 | -1.1239 | | | 0.03577 | | |  |
| ## | Bgsng7131 | -0.3477 | | | -0.04017 | | |  |
| ## | Bgsng7142 | 0.5960 | | | -0.06901 | | |  |
| ## | Bgsng7149 | -0.0745 | | | 0.02303 | | |  |
| ## | Bgsng7150 | 0.6207 | | | -0.01086 | | |  |
| ## | Bgsng7155 | 0.2284 | | | -0.07241 | | |  |
| ## | Bgsng7173 | -0.9953 | | | 0.02941 | | |  |
| ## | Bgsng7177 | 0.4226 | | | -0.07760 | | |  |
| ## | Bgsng7178 | -1.1764 | | | 0.07566 | | |  |
| ## | Bgsng7192 | -0.9135 | | | -0.00481 | | |  |
| ## | Bgsng7202 | 0.5255 | | | 0.00144 | | |  |
| ## | Bgsng7204 | 0.1668 | | | 0.01788 | | |  |
| ## | Bgsng8064 | 1.1785 | | | 0.08485 | | |  |
| ## | Bgsng8169 | 1.1933 | | | 0.03230 | | |  |
| ## | Bgtw1.T1 | 0.3700 | | | -0.04224 | | |  |
| ## | Bgtw1.T2 | 0.7327 | | | -0.09384 | | |  |
| ## | Bgtw10.T1 | 1.4475 | | | -0.10310 | | |  |
| ## | Bgtw10.T2 | 0.2469 | | | 0.02256 | | |  |
| ## | Bgtw11.T1 | 0.2106 | | | -0.09735 | | |  |
| ## | Bgtw11.T2 | 0.6967 | | | -0.06167 | | |  |
| ## | Bgtw12.T1 | 0.1434 | | | -0.19251 | | |  |
| ## | Bgtw12.T2 | 0.6882 | | | -0.17715 | | |  |
| ## | Bgtw2.T1 | -2.0567 | | | 0.02965 | | |  |
| ## | Bgtw2.T2 | -0.9605 | | | -0.02606 | | |  |
| ## | Bgtw3.T1 | 1.6332 | | | -0.02464 | | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| ## | Bgtw3.T2 | 0.5931 | 0.02242 |
| ## | Bgtw4.T1 | -0.0146 | -0.03925 |
| ## | Bgtw4.T2 | 0.3240 | -0.07395 |
| ## | Bgtw4.T3 | 0.5405 | -0.00948 |
| ## | Bgtw5.T1 | -2.0004 | 0.06607 |
| ## | Bgtw5.T2 | 0.2573 | -0.10866 |
| ## | Bgtw6.T1 | -0.4264 | 0.40051 |
| ## | Bgtw6.T2 | -1.2643 | 0.41722 |
| ## | Bgtw7.T1 | 0.2326 | -0.11179 |
| ## | Bgtw7.T2 | -0.6600 | -0.06444 |
| ## | Bgtw8.T1 | 1.5170 | 0.03427 |
| ## | Bgtw8.T2 | 0.8968 | -0.00457 |
| ## | Bgtw9.T1 | -1.3654 | 0.08765 |
| ## | Bgtw9.T2 | -1.7823 | 0.15036 |
| ## |  |  |  |

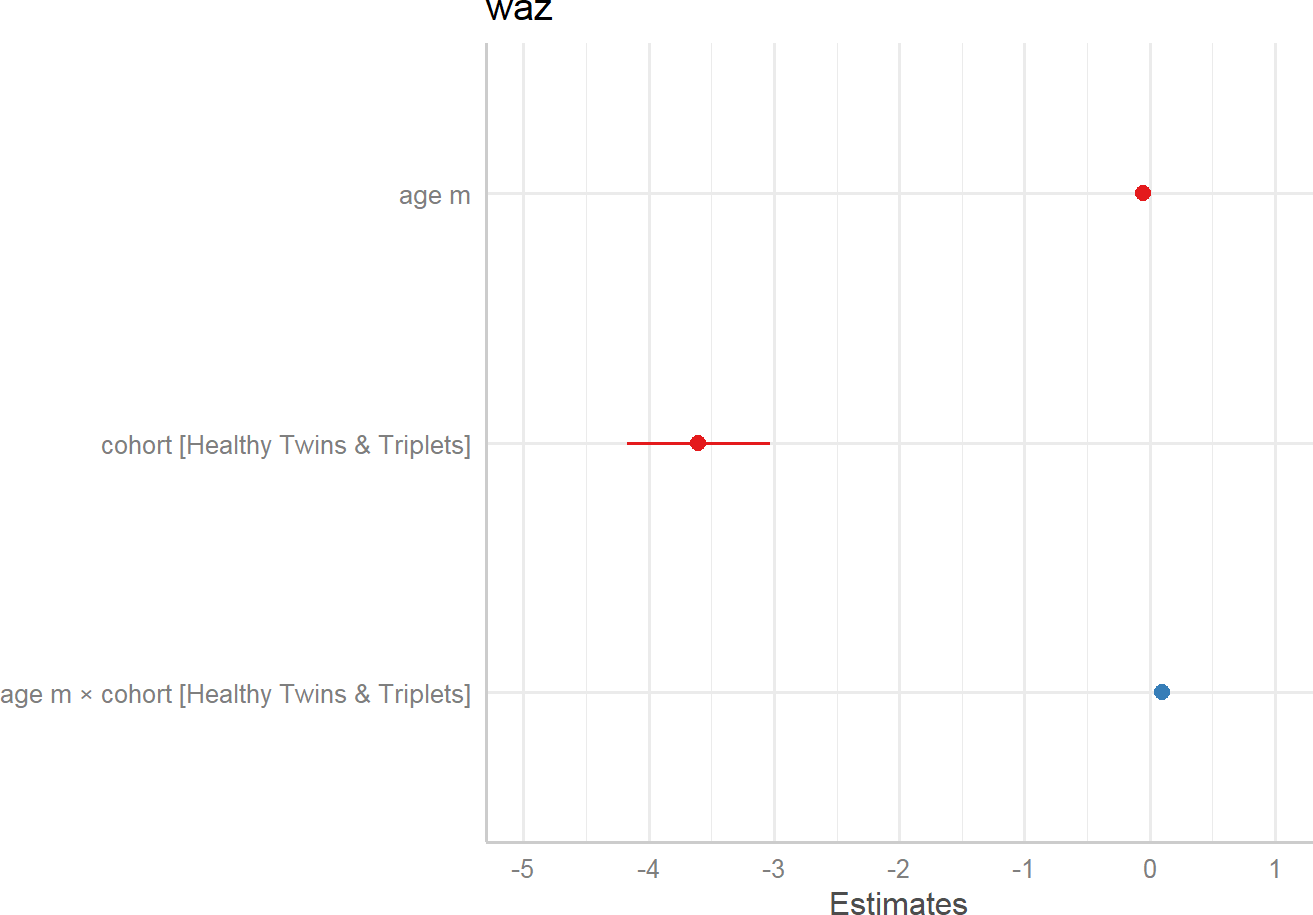
## with conditional variances for "child.id"

#### Visualization of Results

We can visualize with three types of plots: Coefficients, Marginal Effects, Model Diagnostics using *plot\_model()* (with specified *type*) of the package *“sjPlot”* [54] in combination with “*ggplot2*”. Other packages are also useful for plotting results such as *“sjlabelled”* [55]*, “sjmisc”* [56].

Coefficients

Estimates: the plot shows the fixed-effects coefficients and their confidence intervals.



*#install.packages("glmmTMB") #library(**sjPlot)*

*#library(sjlabelled)*

*#library(sjmisc)*

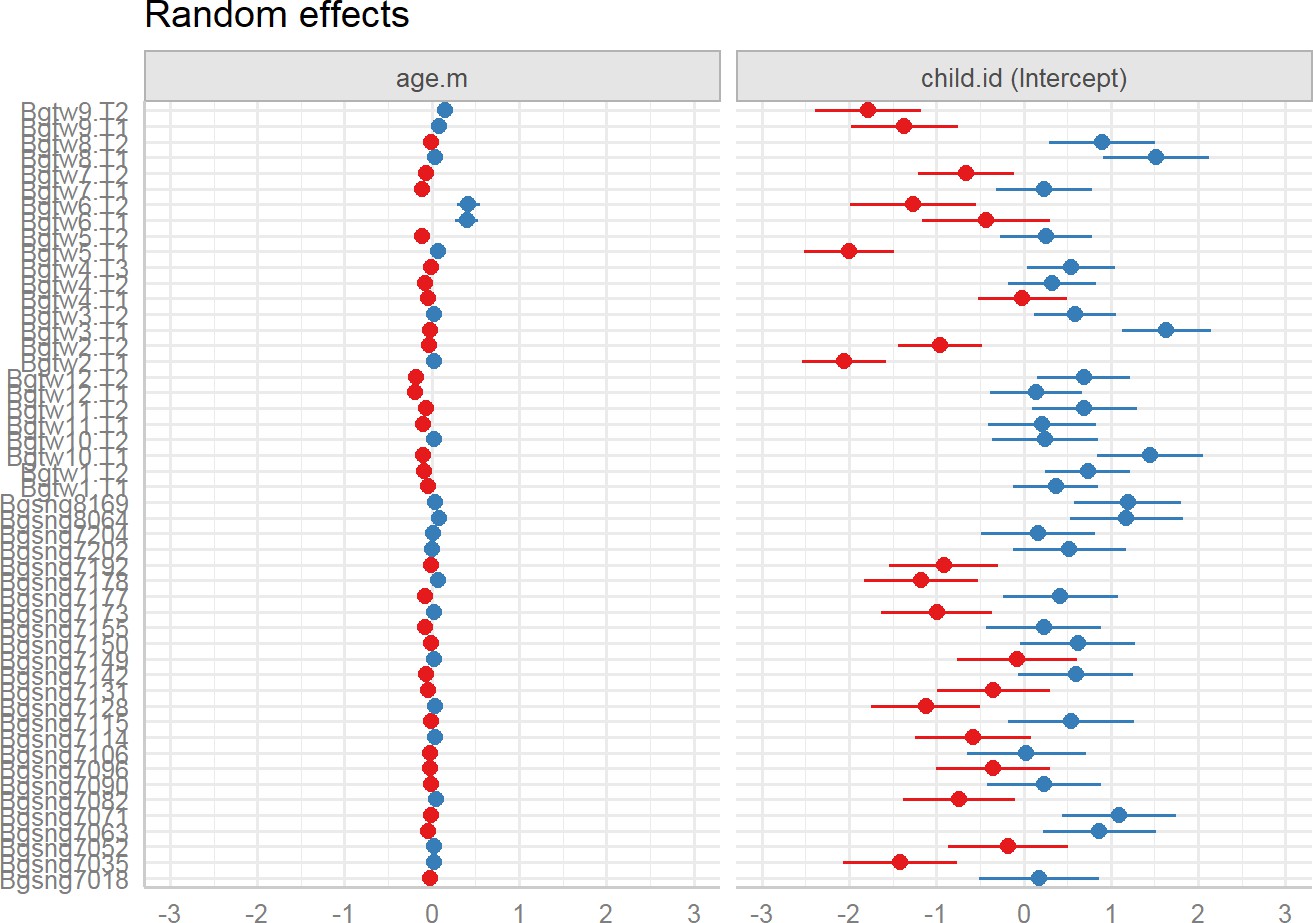
*#library(ggplot2)* theme\_set(theme\_sjplot()) plot\_model(r.s)

Source: Figure by author(s).

**Figure 80**. Coefficient estimates.

Random effects (*type = “re”*): the plot shows the estimated intercept (right panel) and slope (left panel) deviation for every single child in the study.

plot\_model(r.s, type = "re")



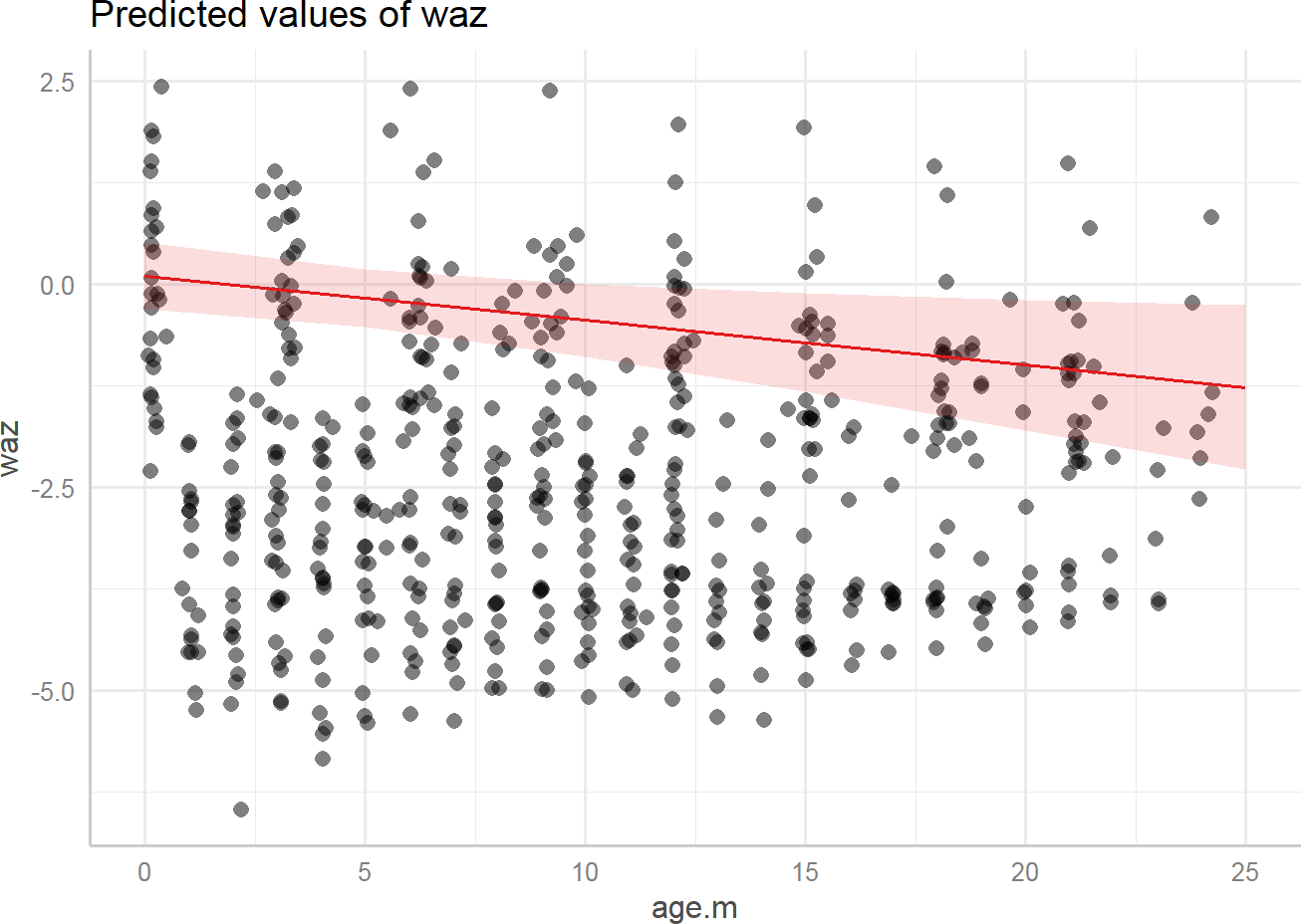
Source: Figure by author(s).

**Figure 81**. Random effects.

Marginal effects

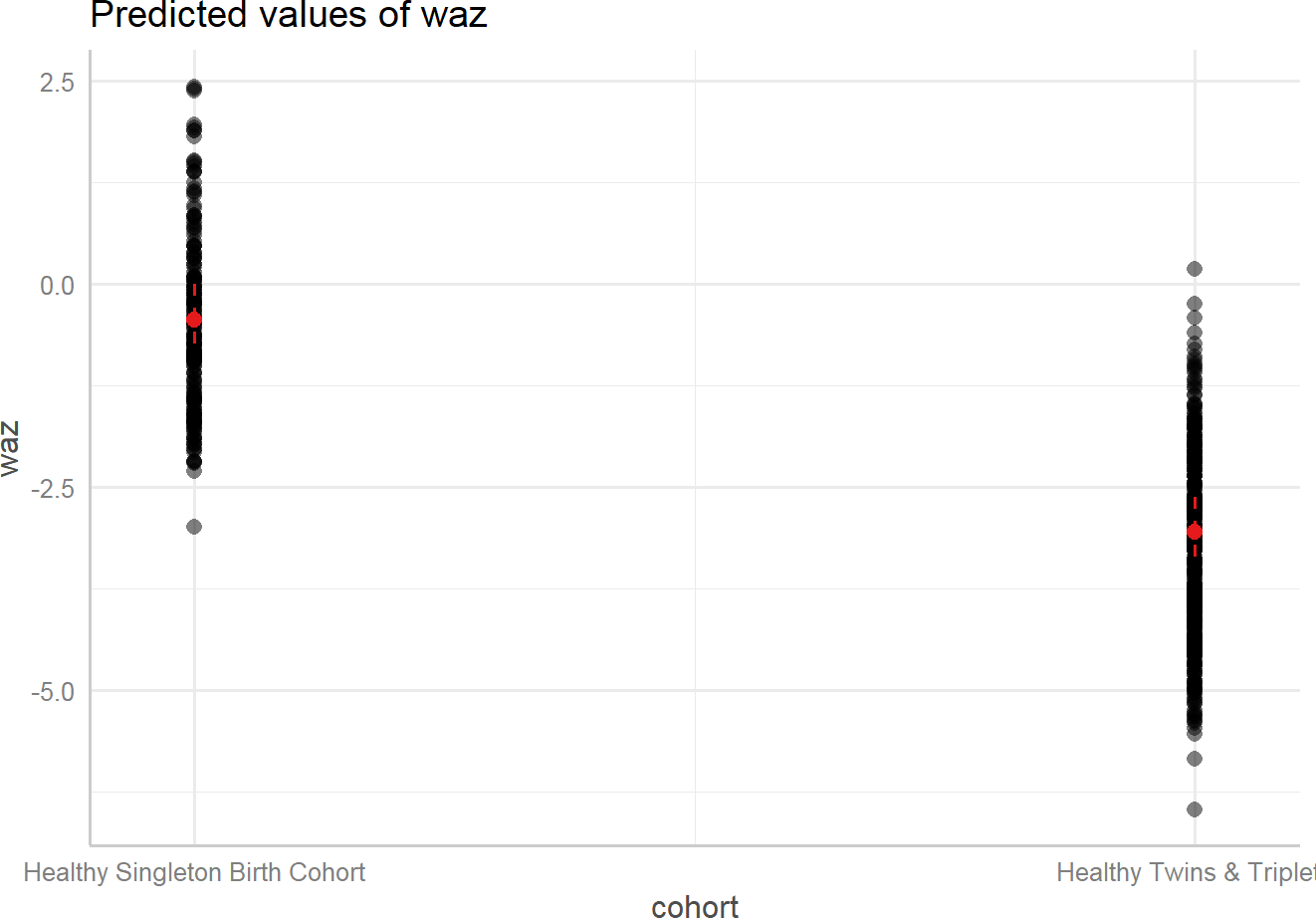
The plot shows predicted values (marginal effects) for specific model terms (*type= “pred”*):

plot\_model(r.s, type = "pred", show.data = T)



Source: Figure by author(s).

**Figure 82**. Predicted values.

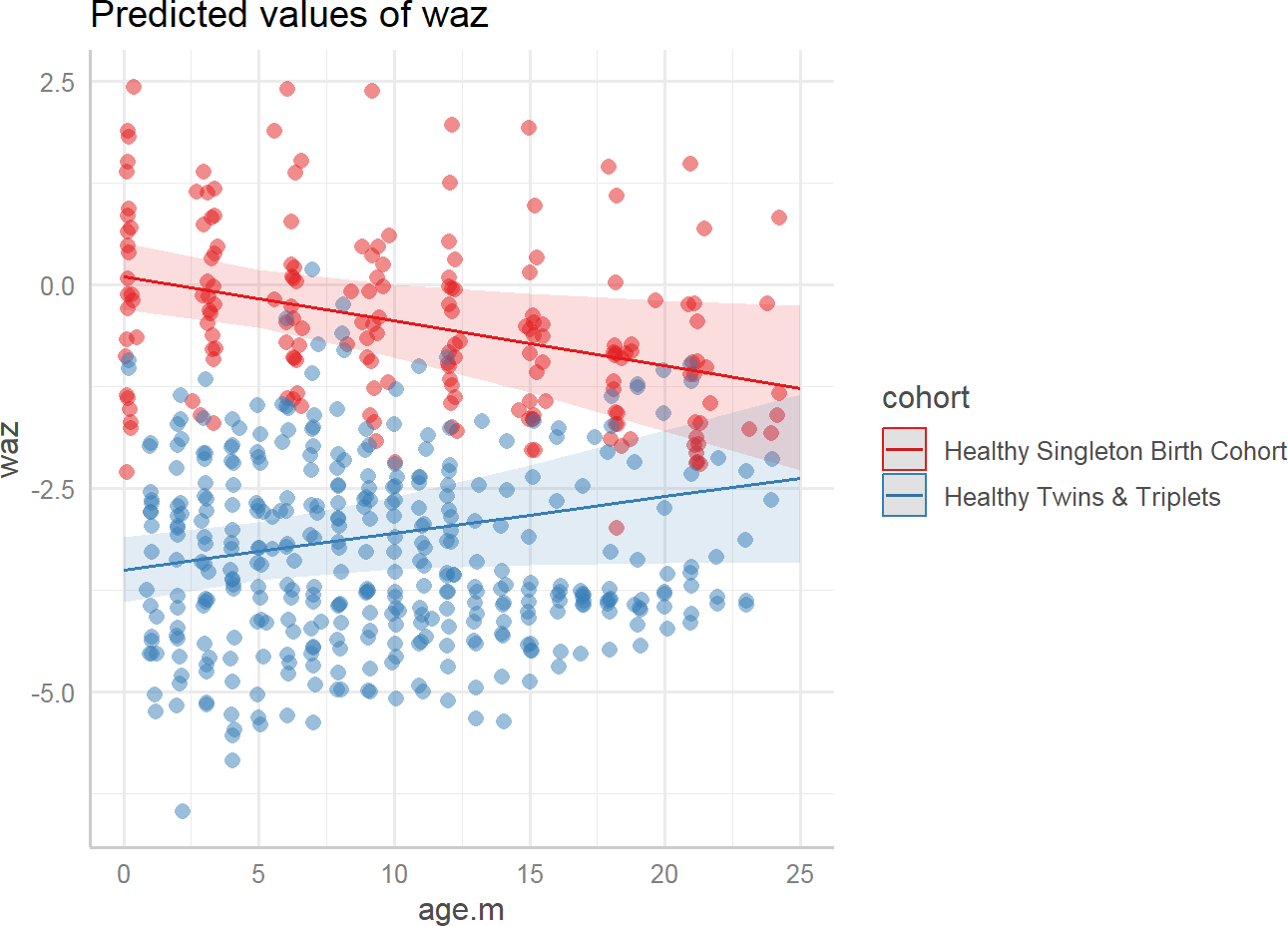


Source: Figure by author(s).

**Figure 83**. Predicted values.

Marginal effects of interaction terms (*type = “int”*): This visualizes the fixed-effect interaction, showing the average predicted slope for the "Singleton" cohort and the average predicted slope for the "Twins & Triplets" cohort, along with the raw data.

plot\_model(r.s, type = "int", show.data = T)



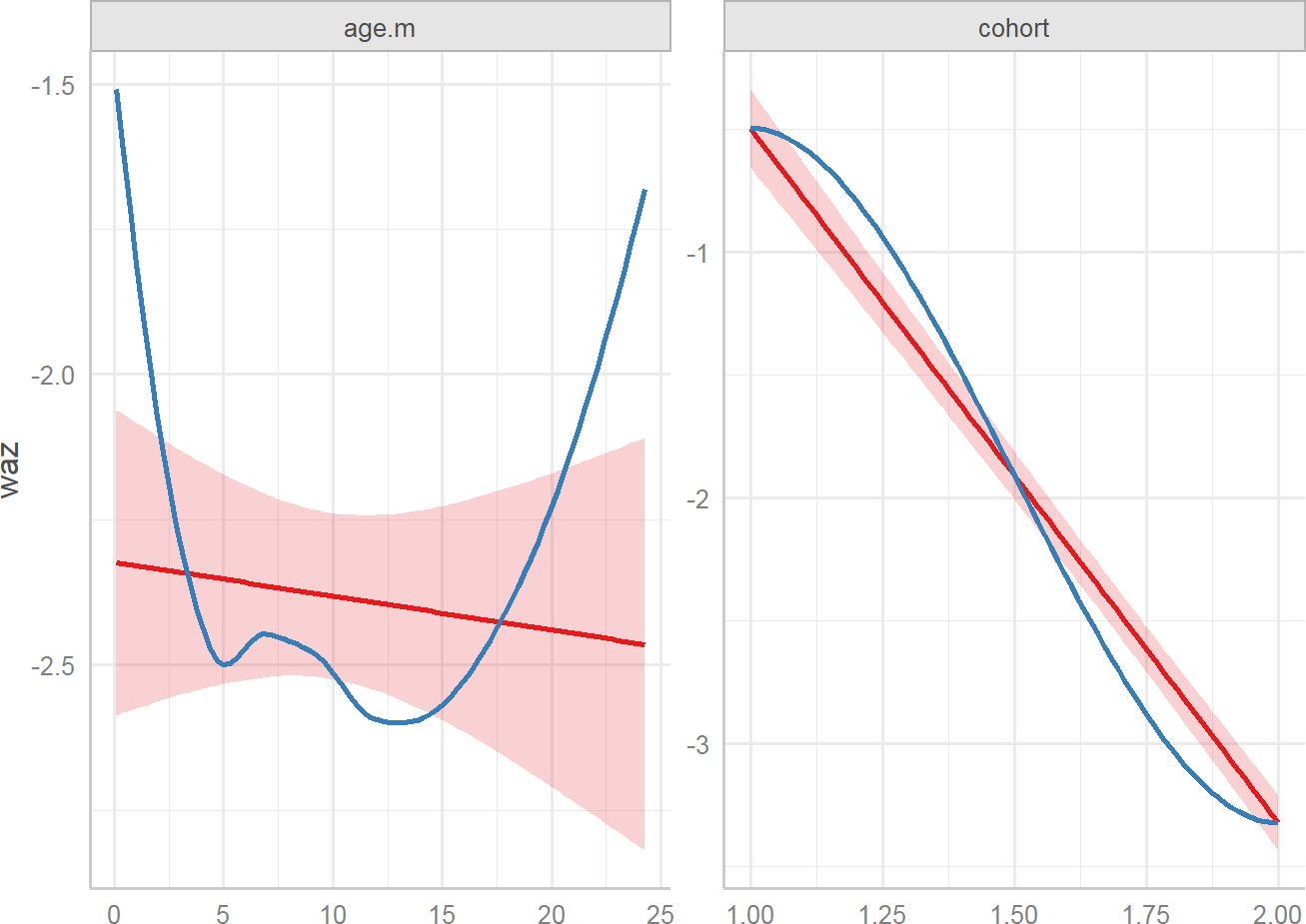
Source: Figure by author(s).

**Figure 84**. Interaction

Model diagnostics

The plot shows the slope of coefficients for each single predictor, against the response (linear relationship between each model term and response) (*type = “slope”*).

plot\_model(r.s, type = "slope")

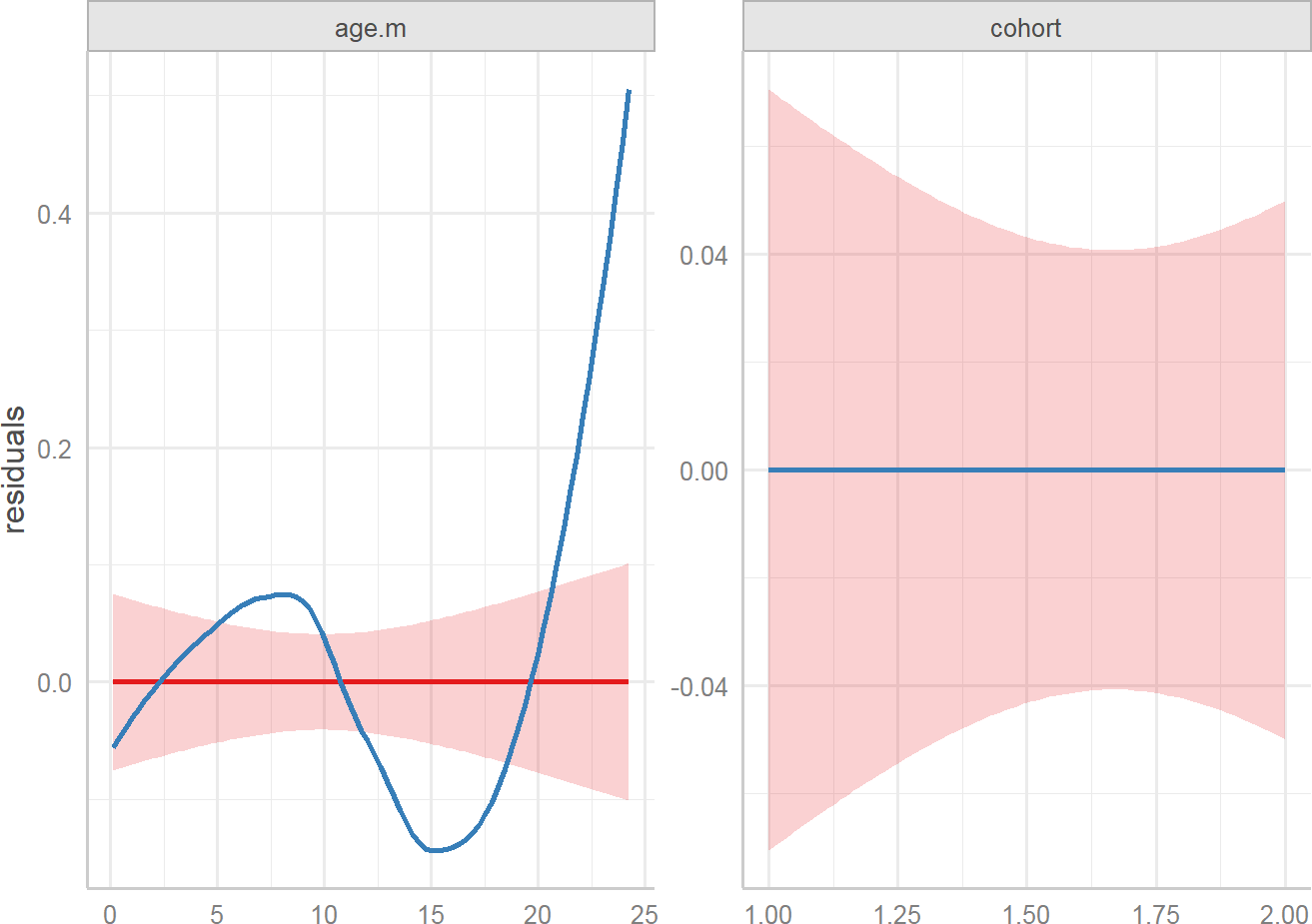


Source: Figure by author(s).

**Figure 85**. Slope.

The plot shows the slope of coefficients for each single predictor, against the residuals (linear relationship between each model term and residuals) (*type = “resid”*).

plot\_model(r.s, type = "resid")



Source: Figure by author(s).

**Figure 86**. Residuals.

Summary of Mixed Model Results as a Table:

*# tab\_model(r.s)*

### GLMM Examples

GLMM may be fitted using *glmer()* function with *link* and *family* specification.

#### Continuous Outcome (Gaussian GLMM)

The below example uses the same Bangladesh data as above. *family = gaussian(link = "identity")* specifies model for continuous outcome.

r.s.glmm<-glmer(waz ~ age.m\*cohort+ (age.m | child.id),

data = waz.nona,

family = gaussian(link = "identity")) summary(r.s.glmm)

## Linear mixed model fit by REML ['lmerMod']

## Formula: waz ~ age.m \* cohort + (age.m | child.id)

## Data: waz.nona

##

## REML criterion at convergence: 1239

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -2.998 -0.510 0.007 0.525 4.531

##

## Random effects:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | Groups | Name | Variance | Std.Dev. Corr |
| ## | child.id | (Intercept) | 0.9505 | 0.975 |
| ## |  | age.m | 0.0126 | 0.112 -0.44 |
| ## | Residual |  | 0.2939 | 0.542 |

## Number of obs: 573, groups: child.id, 50

##

## Fixed effects:

## Estimate Std. Error t value

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | (Intercept) |  |  | 0.1027 | 0.2077 | 0.49 |
| ## | age.m |  |  | -0.0549 | 0.0232 | -2.37 |
| ## | cohortHealthy | Twins | & Triplets | -3.6032 | 0.2913 | -12.37 |
| ## | age.m:cohortHealthy Twins & Triplets 0.0997 0.0332 | | | | | 3.01 |
| ## |  | | | | |  |
| ## | Correlation of Fixed Effects: | | | | |  |
| ## | (Intr) age.m chHT&T | | | | |  |
| ## | age.m -0.475 | | | | |  |
| ## | chrtHlthT&T -0.713 0.339 | | | | |  |
| ## | ag.m:chHT&T 0.332 -0.699 -0.479 | | | | |  |
| ## | optimizer (nloptwrap) convergence code: 0 (OK) | | | | |  |
| ## | Model failed to converge with max|grad| = 0.00410411 | | | | |  |

(tol = 0.002, component 1)

This is equivalent to the LME model using *lmer()* function above.

#### Binary Outcome (Binomial GLMM)

The below example uses a simulated dataset.

* response (a binary 0/1 variable). 1 indicates a patient has achieved a “clinically significant response” based on a depression scale; 0 indicates they have not.
* Research Question: Does the likelihood of a patient responding to treatment change over time, and more importantly, does this change differ between the new Drug and a Placebo?

First, we take a look at our generated dataset:

## 'data.frame': 400 obs. of 8 variables:

## $ subject\_id : Factor w/ 100 levels "1","2","3","4",..: 1 1 1 1 2 2 2 2 3 3 ...

## $ group : Factor w/ 2 levels "Placebo","Drug": 1 1 1 1 1 1 1 1 1 1 ...

## $ rand\_intercept: num -0.841 -0.841 -0.841 -0.841 -0.345 ...

## $ month : int 0 1 2 3 0 1 2 3 0 1 ...

## $ is\_drug : num 0 0 0 0 0 0 0 0 0 0 ...

## $ log\_odds : num -3.34 -3.04 -2.74 -2.44 -2.85 ...

## $ probability : num 0.0342 0.0456 0.0606 0.0801 0.0549 ...

## $ response : int 0 1 0 0 0 0 0 0 0 1 ...

Now, we will analyze this data with *glmer()*. *family = binomial(link = "logit")* tells *glmer()* that the outcome is binary and that it should model the “logit” (log-odds) of a successful response.

*# Fit the Binomial GLMM with a random intercept for subject*

*# This accounts for patients having different baseline propensities to respond.*

glmm\_binary <- glmer(response ~ month \* group + (1 | subject\_id), data = long\_df\_binary,

family = binomial(link = "logit"))

*# Display the summary*

summary(glmm\_binary)

## Generalized linear mixed model fit by maximum likelihood

## (Laplace Approximation) [glmerMod]

## Family: binomial ( logit )

## Formula: response ~ month \* group + (1 | subject\_id)

## Data: long\_df\_binary

##

## AIC BIC logLik -2\*log(L) df.resid

## 410 430 -200 400 395

##

## Scaled residuals:

## Min 1Q Median 3Q Max

## -1.814 -0.454 -0.333 0.551 2764

##

## Random effects:

## Groups Name Variance Std.Dev.

## subject\_id (Intercept) 0.7 0.837

## Number of obs: 400, groups: subject\_id, 100

##

## Fixed effects:

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) -2.250 0.410 -5.48 0.000000042 \*\*\*

## month 0.157 0.188 0.84 0.4035

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | groupDrug | 0.563 | 0.508 | 1.11 | 0.2686 |
| ## | month:groupDrug | 0.767 | 0.253 | 3.03 | 0.0024 \*\* |
| ## | --- |  |  |  |  |
| ## ## | Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' | | | | 0.05 '.' 0.1 ' ' 1 |
| ## | Correlation of Fixed Effects: | | | |  |
| ## | (Intr) month grpDrg | | | |  |
| ## | month -0.750 | | | |  |
| ## | groupDrug -0.735 0.598 | | | |  |
| ## | mnth:grpDrg 0.475 -0.734 -0.753 | | | |  |

The output shows coefficients on Log-Odds Scale so we exponentiate the coefficients to get Odds Ratios.

*# Calculate odds ratios*

exp(fixef(glmm\_binary))

|  |  |  |  |
| --- | --- | --- | --- |
| ## | (Intercept) | month | groupDrug month:groupDrug |
| ## | 0.105 | 1.170 | 1.755 2.153 |

The output shows that:

* OR for “month” (Placebo slope) is 1.17. For patients in the Placebo group, the odds of having a successful response multiply by 1.17 each month. This is statistically significant (p = 0.023).
* OR for “groupDrug” (Baseline difference) is 1.755. At Month 0, the odds of

response for a patient in the Drug group were 1.755 times the odds for a patient in the Placebo group. This difference was not statistically significant.

* OR for “month:groupDrug” (Interaction) is 2.153. This is the most important

result. It’s a ratio of odds ratios. It means that the monthly increase in odds for the Drug group is 2.153 times greater than the monthly increase in odds for the Placebo group. This interaction is highly significant.

Calculating the Slope for the Drug Group: The odds of response for the Drug group multiply by 1.17 \* 2.153 = 2.519 times each month.

#### Count Outcome (Poisson GLMM)

This is used for outcomes that are counts of events (e.g., number of lesions, seizures, hospital visits). We will use the dataset *epil* from the “*MASS”* package [57].

* Outcome Variable: y (the number of epileptic seizures).
* Research Question: Does a new drug (treat) reduce the rate of seizures, accounting for baseline seizure count (base), time (visit), and patient-specific proneness to seizures?

*family = poisson(...)* and *offset = log(period)* are used to model the rate of seizures, just as in the GEE example above.

*# Load necessary packages #library(lme4)*

*#library(MASS)*

*# Load and prepare data*

data(epil) str(epil)

## 'data.frame': 236 obs. of 9 variables:

## $ y : num 5 3 3 3 3 5 3 3 2 4 ...

## $ trt : Factor w/ 2 levels "placebo","progabide": 1 1 1 1 1 1 1 1 1 1 ...

## $ base : int 11 11 11 11 11 11 11 11 6 6 ...

## $ age : int 31 31 31 31 30 30 30 30 25 25 ...

## $ V4 : int 0 0 0 1 0 0 0 1 0 0 ...

## $ subject: int 1 1 1 1 2 2 2 2 3 3 ...

## $ period : int 1 2 3 4 1 2 3 4 1 2 ...

## $ lbase : num -0.756 -0.756 -0.756 -0.756 -0.756 ...

## $ lage : num 0.1142 0.1142 0.1142 0.1142 0.0814 ...

*# The 'period' variable is the duration over which seizures were counted.*

*# It is essential to include this as an offset to model the RATE of seizures.*

epil$age\_num <- as.numeric(epil$age) *# Treat age as a numeric predictor*

*# Fit the Poisson GLMM*

glmm\_poisson <- glmer(y ~ base + trt + age+

(1 | subject), *#random intercept*

data = epil,

family = poisson(link = "log"), *#Poisson distribution .*

offset = log(period)) *# offset to model the rate*

*# Display the summary*

summary(glmm\_poisson)

## Generalized linear mixed model fit by maximum likelihood

## (Laplace Approximation) [glmerMod]

## Family: poisson ( log )

## Formula: y ~ base + trt + age + (1 | subject)

## Data: epil

## Offset: log(period)

##

## AIC BIC logLik -2\*log(L) df.resid

|  |  |  |  |
| --- | --- | --- | --- |
| ## | 1956 1974 | -973 | 1946 231 |
| ## |  |  |  |
| ## | Scaled residuals: |  |  |
| ## | Min 1Q Median | 3Q | Max |

## -5.294 -1.093 -0.066 1.290 13.015

##

## Random effects:

## Groups Name Variance Std.Dev.

## subject (Intercept) 0.286 0.534

## Number of obs: 236, groups: subject, 59

##

## Fixed effects:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## | Estimate | Std. Error | z | value | Pr(>|z|) | |  |
| ## (Intercept) | -0.3942 | 0.4072 |  | -0.97 | 0.33 | |
| ## base | 0.0272 | 0.0028 |  | 9.73 | <0.0000000000000002 | | \*\*\* |
| ## trtprogabide | -0.2611 | 0.1541 |  | -1.69 | 0.09 | | . |
| ## age | 0.0143 | 0.0126 |  | 1.14 | 0.26 | |  |
| ## --- |  |  |  |  |  | |  |
| ## Signif. codes: 0 '\*\*\*' 0.001 '\*\*'  ## | | | | 0.01 '\*' | | 0.05 '.' 0.1 ' ' | 1 |
| ## Correlation of Fixed Effects: | | | |  | | | |
| ## (Intr) base trtprg | | | |
| ## base -0.395 | | | |
| ## trtprogabid -0.291 -0.006 | | | |
| ## age -0.937 0.190 0.114 | | | |

The output shows the coefficients on log scale so we exponentiate the coefficients to get Incident Rate Ratios (IRRs):

*# Calculate Incident Rate Ratios*

exp(fixef(glmm\_poisson))

|  |  |  |
| --- | --- | --- |
| ## (Intercept) | base trtprogabide | age |
| ## 0.674 | 1.028 0.770 | 1.014 |

The output shows that IRR for “trtprogabide” is 0.77. The rate of seizures for patients on the new drug (“progabide”) is 0.77 of the rate for patients on the placebo. In other words, the treatment is associated with a 0.23 reduction in the seizure, though this effect is not statistically significant.

## GEE or Mixed Effect Model

The two approaches have different targets for inferences and address subtly different questions about longitudinal change

* GEE: Marginal (population-average): address the correlation among repeated measurements by robust variance estimation
* Mixed effect: Conditional (subject-specific): provide an explanation for the source of correlation at different levels

Below are example scenarios for choosing GEE or GLME.

Scenario 1: Public Health Policy

* Question: Does a city-wide smoking ban reduce the overall prevalence of respiratory illness over 5 years?
* Choice: GEE. The interest is in the effect on the entire city’s population, not on predicting a specific individual’s illness trajectory.

Scenario 2: Child Development Study

* Question: What is the growth trajectory of reading ability in children from ages 6 to 10? Are the starting ability (intercept) and the rate of learning (slope) correlated?
* Choice: Mixed effect. The focus is explicitly on modeling the individual growth

curves and understanding the sources of variation between children.

## Useful Resources

### Courses

* + - * SISCR Course 2016: “Introduction to Longitudinal Data Analysis” by Benjamin French, PhD (University of Pennsylvania) and Colleen Sitlani, PhD (University of Washington).

### Online Tutorials

* + - * [Fitting Linear Mixed-Effects Models Using lme4](https://www.jstatsoft.org/article/view/v067i01) [37].
      * [Introduction To Generalized Linear Mixed Models](https://stats.idre.ucla.edu/other/mult-pkg/introduction-to-generalized-linear-mixed-models/) (https://stats.idre.ucla.edu/other/mult-pkg/introduction-to-generalized-linear-mixed-models/).
      * [Generalized Estimating Equations (GEE)](https://rlbarter.github.io/Practical-Statistics/2017/05/10/generalized-estimating-equations-gee/) (https://rlbarter.github.io/Practical-Statistics/2017/05/10/generalized-estimating-equations-gee/)
      * [To GEE or not to GEE](https://journals.lww.com/epidem/Fulltext/2010/07000/To_GEE_or_Not_to_GEE__Comparing_Population_Average.7.aspx) [60].

### Books, Other Resources

* + - * [APPLIED LONGITUDINAL DATA ANALYSIS](https://stats.idre.ucla.edu/other/examples/alda/) [58].
      * [Linear Mixed-Effects Models Using R](https://link.springer.com/book/10.1007/978-1-4614-3900-4) [59].

## Chapter Summary

* *“lme4”* and “*geepack”* are the main package for longitudinal data analysis
* *geeglm()* function to fit GEE model which focuses on population-average and

address correlation of repeated measures as nuisance

* + *family = gaussian*: for continuous outcome, *family = binomial* for binary outcome, *family = poisson* with *offset = log(time)* for count outcome
  + *corstr*: for correlation structure (*QIC()* to select the best correlation structure)
* *lmer()* function to fit Linear Mixed Effect (LME) models which include:
  + Fixed Effects: population-average relationship between a predictor and the

outcome

* + Random effects: heterogeneity across subjects (as random intercepts (*1 | subject\_id*) and random slopes (*time | subject\_id*))
* “*sjPlot*”, “*sjlabelled*”, “*sjmisc*” are packages to help visualize LME model results
* *glmer()* function to fit Generalized Linear Mixed-Effects Models (GLMM) (an

extension of the Linear Mixed-Effects (LME) model) that handle various types of outcomes.

* + *family = gaussian(link = "identity"):* for continuous outcomes
  + *family = binomial(link = "logit"):* for binary outcomes
  + *family = poisson(link = "log")* with *offset = log(time)*: for count

outcomes

# Analysis of Large Consortium Data with R

In bio-medical research, especially in epidemiology, it is common for research groups from around the world to pool their data into a large consortium. A consortium combines multiple individual studies to create a single, massive dataset. This approach has several key advantages: 1) Increased Statistical Power: By dramatically increasing the total sample size, consortia can detect small but important risk factors that individual studies would miss; 2) Ability to Study Rare Outcomes: For diseases like liver cancer, which are rare in the general population, individual cohorts may only have a small number of cases. Pooling data aggregates thousands of cases, making meaningful analysis possible; 3) Diverse Populations: Consortia often include data from different countries and ethnicities, allowing researchers to see if findings are consistent across diverse populations or if they are unique to a specific group.

Analyzing consortium data, however, is more complex than analyzing a single study. One must account for the fact that the data comes from different sources, which may have different baseline disease rates, follow-up procedures, and population characteristics.

This chapter will guide through the analyses to generate similar results to those of the paper “Diabetes is associated with increased liver cancer incidence and mortality in adults: A report from Asia Cohort Consortium” published in [International Journal of Cancer](https://doi.org/10.1002/ijc.34965) by Nhan Thi Ho et al [61]. Using time-to-event (survival) analysis as the main analysis example, the chapter will provide examples for: 1) Pooled Analysis: Analyzing all data together while using stratification to account for differences between studies; 2) Study-Specific Analysis: Running the analysis separately within each individual study; 3) Meta-Analysis: Formally combining the results from the individual studies to get a robust, summary estimate; 4) Sensitivity Analysis: Checking if the overall result is overly influenced by any single study.

This chapter will provide examples for practical R skills including writing R functions, loops to perform repeated analysis more efficiently and provide R code examples to generate clear, report-ready tables and plots. This chapter focuses on how to do specific analyses and result display using R, not on result output interpretation.

## About Data

The example dataset used in this chapter was generated to mimic the real data used in the above mentioned paper. The generated example dataset includes data from multiple cohort studies (identified by “studycode”) regarding the association between a certain type of cancer (“cancer”), time to even (“ttcancer”) and a main exposure of interest (“main.exposure”) as well as several covariates like age, gender, BMI, etc.

### Load Required Packages and Example Data

The example dataset is hosted in Github: *load(url(...))* will loads the .rda (R Data) file from the provided GitHub URL into our R environment. The data is stored in an object named “candat”. The main R packages used in this chapter include *"lubridate"* [45]*,"survival"* [43]*,"cmprsk"* [47]*, "survminer"* [44]*,"gridExtra"* [62]*,"Hmisc"* [63]*, "psych"* [64]*, "gmodels"* [65]*, "gtsummary"* [30]*, "meta"* [66]*, "patchwork"* [67]*, "rio"* [14]*,"arsenal"* [19]. Other supporting packages ("*knitr*" [5], "*rmarkdown*" [6], "*kableExtra*" [12], "*tidyverse*" [11]) are used routinely as in previous chapters. Now, we load the packages and the example data.

*#load multiple packages*

Packages <- c("knitr","rmarkdown","kableExtra","tidyverse", "rio","arsenal", "lubridate","survival","cmprsk", "survminer","gridExtra","Hmisc",

"psych", "gmodels", "gtsummary", "meta", "patchwork")

lapply(Packages, library, character.only = TRUE)

*# Data path*

url\_base <- "https://raw.githubusercontent.com/nhanhocu/"

data\_path <- "biodata-r/main/candat.rda"

data\_url <- paste0(url\_base, data\_path)

*#load data*

con <- url(data\_url) load(con)

close(con)

### Summary of data

The first step is to examine the data by summarizing the main variables by groups.

#### Summary of data overall and by studies

We use the *tableby()* function from the “*arsenal”* package to create a comprehensive descriptive table, showing the characteristics of each study side-by-side with the overall total.

First, we define summary options and make summary table:

*# Define summary options*

my\_controls<-tableby.control( test = F,

total = T,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = c("meansd","medianq1q3","range","Nmiss2"),

cat.stats = c("countpct","Nmiss2"),

stats.labels = list(

meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)",

range = "Min-Max",

Nmiss2 = "Missing"

)

)

sumvar<-c("cancer.f", "ttcancer", "main.exposure",

"age", "age.cat", "gender",

"bmi", "bmi.cat",

"smoking", "alcoholuse")

mylabels <-as.list(c("Cancer",

"Time-to-cancer (year)",

"Exposure of interest",

"Age (year)", "Age categories",

"Gender",

"BMI", "BMI categories",

"Smoking", "Alcohol use"))

names(mylabels)<-sumvar

tab1 <- tableby(as.formula(paste("studycode",

paste(sumvar,collapse = "+"),

sep = "~")),

data = candat,

control = my\_controls)

kable(summary(tab1,

labelTranslations = mylabels,

text = TRUE),

caption = "Summary of data by study and overall")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 10**. Summary of data by study and overall.

175

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Study1 (N = 33,529)** | **Study2 (N = 31,552)** | **Study3 (N = 20,636)** | **Study4 (N = 42,751)** | **Study5 (N = 47,605)** | **Study6 (N = 31,345)** | **Study7 (N = 51,252)** | **Study8 (N = 52,883)** | **Study9 (N = 63,257)** | **Total (N = 374,810)** |
| Cancer |  |  |  |  |  |  |  |  |  |  |
| - No | 33,325 (99.4%) | 31,397 (99.5%) | 20,332 (98.6%) | 39,528 (99.6%) | 47,134 (99.0%) | 31,229 (99.6%) | 50,825 (99.2%) | 52,145 (98.6%) | 62,736 (99.2%) | 368,651 (99.2%) |
| - Yes | 204 (0.6%) | 155 (0.5%) | 299 (1.4%) | 150 (0.4%) | 471 (1.0%) | 116 (0.4%) | 427 (0.8%) | 738 (1.4%) | 521 (0.8%) | 3081 (0.8%) |
| - Missing | 0 | 0 | 5 | 3073 | 0 | 0 | 0 | 0 | 0 | 3078 |
| Time-to-cancer (year) |  |  |  |  |  |  |  |  |  |  |
| - Mean (SD) | 11.508 (5.134) | 13.507 (4.232) | 13.872 (4.670) | 9.301 (3.276) | 21.435 (6.039) | 7.600 (2.606) | 10.702 (4.304) | 21.889 (10.281) | 13.872 (3.839) | 14.220 (7.378) |
| - Median (Q1, Q3) | 15.165 (7.485, 15.165) | 15.578 (15.140, 15.578) | 13.864 (10.891, 17.443) | 9.254 (6.606, 12.044) | 24.583 (20.931, 24.583) | 8.999 (7.556, 8.999) | 13.246 (9.251, 13.246) | 21.916 (10.916, 32.000) | 15.165 (12.591, 16.657) | 13.246 (8.999, 16.925) |
| - Min−Max | 0.000−15.502 | 0.000−15.578 | 0.027−21.804 | 0.022−15.403 | 0.038−24.583 | 0.016−8.999 | 0.000−13.246 | 0.000−38.916 | 0.008−17.892 | 0.000−38.916 |
| - Missing | 0 | 0 | 5 | 3073 | 0 | 0 | 0 | 3055 | 0 | 6133 |
| Exposure of interest |  |  |  |  |  |  |  |  |  |  |
| - No | 22,048 (95.3%) | 30,115 (95.4%) | 18,224 (95.4%) | 39,033 (93.8%) | 45,566 (95.7%) | 29,558 (94.3%) | 47,845 (93.4%) | 40,766 (92.8%) | 57,561 (91.0%) | 330,716 (93.7%) |
| - Yes | 1076 (4.7%) | 1437 (4.6%) | 885 (4.6%) | 2559 (6.2%) | 2039 (4.3%) | 1787 (5.7%) | 3407 (6.6%) | 3181 (7.2%) | 5696 (9.0%) | 22067 (6.3%) |
| - Missing | 10,405 | 0 | 1527 | 1159 | 0 | 0 | 0 | 8936 | 0 | 22,027 |
| Age (year) |  |  |  |  |  |  |  |  |  |  |
| - Mean (SD) | 56.424 (11.408) | 55.959 (12.889) | 54.118 (14.320) | 50.009 (9.224) | 52.141 (7.538) | 57.324 (11.345) | 60.511 (10.302) | 52.249 (13.684) | 56.505 (8.016) | 55.064 (11.227) |
| - Median (Q1, Q3) | 54.000 (47.000, 64.000) | 54.667 (44.854, 65.087) | 56.000 (45.000, 64.000) | 50.000 (43.000, 56.000) | 53.000 (45.000, 59.000) | 56.000 (48.000, 65.000) | 62.000 (53.000, 68.000) | 51.400 (41.800, 61.800) | 56.000 (50.000, 63.000) | 55.000 (47.000, 63.000) |
| - Min−Max | 40.000−103.000 | 35.425−101.051 | 15.000−91.000 | 16.000−85.000 | 40.000−64.000 | 40.000−98.000 | 40.000−80.000 | 19.300−98.700 | 43.000−83.000 | 15.000−103.000 |
| - Missing | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 3055 | 0 | 3060 |
| Age categories |  |  |  |  |  |  |  |  |  |  |
| - <40 years old | 0 (0.0%) | 3123 (9.9%) | 3188 (15.5%) | 5373 (12.6%) | 0 (0.0%) | 0 (0.0%) | 0 (0.0%) | 10,122 (20.3%) | 0 (0.0%) | 21,806 (5.9%) |
| - > = 40&<60 years old | 21,564 (64.3%) | 16,781 (53.2%) | 8931 (43.3%) | 30,091 (70.4%) | 37,138 (78.0%) | 19,140 (61.1%) | 21,097 (41.2%) | 25,069 (50.3%) | 40,852 (64.6%) | 220,663 (59.4%) |
| - > = 60 years old | 11,965 (35.7%) | 11,648 (36.9%) | 8512 (41.3%) | 7287 (17.0%) | 10,467 (22.0%) | 12,205 (38.9%) | 30,155 (58.8%) | 14,637 (29.4%) | 22,405 (35.4%) | 129,281 (34.8%) |
| - Missing | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 3055 | 0 | 3060 |
| Gender |  |  |  |  |  |  |  |  |  |  |
| - Female | 17,783 (53.0%) | 17,125 (54.3%) | 12,399 (60.1%) | 21,326 (49.9%) | 24,769 (52.0%) | 17,353 (55.4%) | 26,679 (52.1%) | 32,493 (61.4%) | 35,303 (55.8%) | 205,230 (54.8%) |
| - Male | 15,746 (47.0%) | 14,427 (45.7%) | 8232 (39.9%) | 21,425 (50.1%) | 22,836 (48.0%) | 13,992 (44.6%) | 24,573 (47.9%) | 20,390 (38.6%) | 27,954 (44.2%) | 169,575 (45.2%) |
| - Missing | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 5 |
| BMI |  |  |  |  |  |  |  |  |  |  |
| - Mean (SD) | 22.101 (2.976) | 22.206 (2.896) | 23.571 (3.285) | 23.859 (2.997) | 23.644 (2.984) | 23.254 (3.443) | 23.532 (3.324) | 35.318 (2912.005) | 23.116 (3.274) | 24.890 (1085.530) |
| - Median (Q1, Q3) | 21.971 (20.047, 23.875) | 22.052 (20.269, 23.957) | 23.415 (21.259, 25.606) | 23.723 (21.778, 25.716) | 23.438 (21.514, 25.433) | 22.959 (21.077, 25.104) | 23.309 (21.403, 25.391) | 21.504 (19.628, 23.805) | 23.112 (21.094, 24.655) | 22.880 (20.830, 24.889) |
| - Min−Max | 10.644−56.960 | 9.934−57.398 | 12.955−63.267 | 11.204−51.554 | 11.687−54.861 | 7.171−64.930 | 8.002−82.305 | −3.556−650000.000 | 9.778−68.966 | −3.556−650000.000 |
| - Missing | 1306 | 1845 | 1620 | 342 | 2737 | 1791 | 3542 | 3055 | 0 | 16,238 |
| BMI categories |  |  |  |  |  |  |  |  |  |  |
| - <23 | 20,706 (64.3%) | 18,760 (63.2%) | 8645 (45.5%) | 17,044 (40.2%) | 19,763 (44.0%) | 14,859 (50.3%) | 21,883 (45.9%) | 0 | 30,366 (48.0%) | 152,026 (49.2%) |
| - > = 23 | 11,517 (35.7%) | 10,947 (36.8%) | 10,371 (54.5%) | 25,365 (59.8%) | 25,105 (56.0%) | 14,695 (49.7%) | 25,827 (54.1%) | 0 | 32,891 (52.0%) | 156,718 (50.8%) |
| - Missing | 1306 | 1845 | 1620 | 342 | 2737 | 1791 | 3542 | 52,883 | 0 | 66,066 |
| Smoking |  |  |  |  |  |  |  |  |  |  |
| - No | 15,285 (49.3%) | 14,927 (50.9%) | 12,819 (63.2%) | 22,339 (54.3%) | 20,139 (50.0%) | 12,704 (57.0%) | 22,310 (51.4%) | 28,408 (56.2%) | 43,930 (69.4%) | 192,861 (56.5%) |
| - Yes | 15,723 (50.7%) | 14,391 (49.1%) | 7471 (36.8%) | 18,799 (45.7%) | 20,103 (50.0%) | 9597 (43.0%) | 21,097 (48.6%) | 22,102 (43.8%) | 19,327 (30.6%) | 148,610 (43.5%) |
| - Missing | 2521 | 2234 | 346 | 1613 | 7363 | 9044 | 7845 | 2373 | 0 | 33,339 |
| Alcohol use |  |  |  |  |  |  |  |  |  |  |
| - No | 10,318 (34.4%) | 0 | 11,754 (58.1%) | 13,442 (32.2%) | 17,255 (41.1%) | 9637 (39.4%) | 19,408 (43.3%) | 32,189 (60.9%) | 51,384 (81.2%) | 165,387 (51.8%) |
| - Yes | 19,682 (65.6%) | 0 | 8462 (41.9%) | 28,341 (67.8%) | 24,767 (58.9%) | 14,803 (60.6%) | 25,423 (56.7%) | 20,694 (39.1%) | 11,873 (18.8%) | 154,045 (48.2%) |
| - Missing | 3529 | 31,552 | 420 | 968 | 5583 | 6905 | 6421 | 0 | 0 | 55,378 |

Source: Table by author(s).

#### Summary of data by Main Exposure Status

Next, we look at the pooled data, comparing the characteristics of the "exposed" (“Yes”) versus "unexposed" (“No’) groups.

*# Define summary options*

my\_controls<-tableby.control( test = F,

total = F,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = c("meansd","medianq1q3","range","Nmiss2"),

cat.stats = c("countpct","Nmiss2"),

stats.labels = list(

meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)",

range = "Min-Max",

Nmiss2 = "Missing"

)

)

sumvar<-c("cancer.f", "ttcancer",

"age", "age.cat", "gender",

"bmi", "bmi.cat",

"smoking","alcoholuse")

mylabels <-as.list(c("Cancer",

"Time-to-cancer (year)",

"Age (year)", "Age categories",

"Gender",

"BMI", "BMI categories",

"Smoking", "Alcohol use"))

names(mylabels)<-sumvar

tab1 <- tableby(as.formula(paste("main.exposure",

paste(sumvar, collapse = "+"),

sep = "~")),

data = candat, control = my\_controls)

kable(summary(tab1,

labelTranslations = mylabels,

text = TRUE),

caption = "Summary of data by main exposure")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 11**. Summary of data by main exposure.

|  |  |  |
| --- | --- | --- |
|  | **No (N = 330716)** | **Yes (N = 22067)** |
| Cancer |  |  |
| - No | 325,309 (99.2%) | 21,524 (98.3%) |
| - Yes | 2523 (0.8%) | 373 (1.7%) |
| - Missing | 2884 | 170 |
| Time-to-cancer (year) |  |  |
| - Mean (SD) | 14.229 (7.177) | 12.264 (6.831) |
| - Median (Q1, Q3) | 13.246 (8.999, 16.975) | 12.493 (7.748, 15.578) |
| - Min−Max | 0.000−38.916 | 0.000−38.916 |
| - Missing | 4667 | 339 |
| Age (year) |  |  |
| - Mean (SD) | 54.725 (11.037) | 59.338 (9.905) |
| - Median (Q1, Q3) | 54.000 (46.500, 62.000) | 60.000 (52.000, 66.000) |
| - Min−Max | 15.000−103.000 | 15.000−94.200 |
| - Missing | 1786 | 169 |
| Age categories |  |  |
| - <40 years old | 18,969 (5.8%) | 422 (1.9%) |
| - > = 40&<60 years old | 200,318 (60.9%) | 10,452 (47.7%) |
| - > = 60 years old | 109,643 (33.3%) | 11,024 (50.3%) |
| - Missing | 1786 | 169 |
| Gender |  |  |
| - Female | 179,833 (54.4%) | 10,069 (45.6%) |
| - Male | 150,880 (45.6%) | 11,998 (54.4%) |
| - Missing | 3 | 0 |
| BMI |  |  |
| - Mean (SD) | 25.130 (1153.341) | 23.857 (3.580) |
| - Median (Q1, Q3) | 22.880 (20.889, 24.889) | 23.508 (21.671, 25.778) |
| - Min−Max | −3.556−650000.000 | 10.644−121.528 |
| - Missing | 13,070 | 831 |
| BMI categories |  |  |
| - <23 | 137,894 (49.5%) | 7134 (39.1%) |
| - > = 23 | 140,769 (50.5%) | 11,090 (60.9%) |
| - Missing | 52,053 | 3843 |

**Table 11.** *Cont.*

|  |  |  |
| --- | --- | --- |
|  | **No (N = 330716)** | **Yes (N = 22067)** |
| Smoking |  |  |
| - No | 170,992 (56.6%) | 10,118 (49.5%) |
| - Yes | 131,083 (43.4%) | 10,341 (50.5%) |
| - Missing | 28,641 | 1608 |
| Alcohol use |  |  |
| - No | 143,094 (51.1%) | 105,32 (54.0%) |
| - Yes | 137,004 (48.9%) | 8954 (46.0%) |
| - Missing | 50,618 | 2581 |

Source: Table by author(s).

## Analysis Approaches

### Estimate Incidence of Cancer by Person Years

We first calculate the crude cancer incidence rate, which is the number of new cancer cases divided by the total "person-years" of follow-up. Incidence of cancer (per 100,000 person\_year) overall and by “main.exposure” status (with *group\_by()*) for pooled data of all studies are calculated as below: *person.year = sum(ttcancer, na.rm = T)*: this code calculates total person-years by summing the follow-up time (*ttcancer*) for all participants; *cancer.incidence = ... \* 100000*: This standardizes the rate to be "per 100,000 person-years," which is a standard epidemiological metric.

*# Overall*

py<- candat %>%

filter(!is.na(main.exposure)) %>%

summarise(person.year = sum(ttcancer, na.rm = T),

cancer.case = sum(as.numeric(as.character(cancer)),

na.rm = T),

cancer.incidence = round((cancer.case/person.year)\*100000,2))

py

## person.year cancer.case cancer.incidence

## 1 4905681 2896 59

*# By main exposure*

py.be<-candat %>%

filter(!is.na(main.exposure)) %>%

group\_by(main.exposure)%>%

summarise(person.year = sum(ttcancer, na.rm = T),

cancer.case = sum(as.numeric(as.character(cancer)), na.rm = T),

cancer.incidence = round((cancer.case/person.year)\*100000,2))

py.be

## # A tibble: 2 x 4

## main.exposure person.year cancer.case cancer.incidence

## <fct> <dbl> <dbl> <dbl>

## 1 No 4639216. 2523 54.4

## 2 Yes 266465. 373 140.

The example below calculates incidence of cancer (per 100,000 person\_year) for each study (with *group\_by(studycode)*) and summarize the results as a table:

inc.study<-candat %>%

filter(!is.na(main.exposure)) %>%

group\_by(studycode)%>%

summarise(py.study = sum(ttcancer, na.rm = T),

cancer = sum(as.numeric(as.character(cancer)), na.rm = T),

cancer.inc = round((cancer/py.study)\*100000,2))

kable(inc.study, caption = "Incidence by study")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 12**. Incidence by study.

|  |  |  |  |
| --- | --- | --- | --- |
| **studycode** | **py.study** | **cancer** | **cancer.inc** |
| study1 | 269,191 | 122 | 45.3 |
| study2 | 426,187 | 155 | 36.4 |
| study3 | 267,830 | 281 | 104.9 |
| study4 | 356,565 | 145 | 40.7 |
| study5 | 1,020,397 | 471 | 46.2 |
| study6 | 238,214 | 116 | 48.7 |
| study7 | 548,492 | 427 | 77.8 |
| study8 | 901,276 | 658 | 73.0 |
| study9 | 877,529 | 521 | 59.4 |

Source: Table by author(s).

The example below calculates incidence of cancer (per 100,000 person\_year) for each study and by main exposure status (with *group\_by(main.exposure, studycode)*) and summarize the results as a table:

inc.ex.study<-candat %>%

filter(!is.na(main.exposure)) %>%

group\_by(main.exposure,studycode)%>%

summarise(py.ex.study = sum(ttcancer, na.rm = T),

cancer = sum(as.numeric(as.character(cancer)), na.rm = T),

cancer.inc = round((cancer/py.ex.study)\*100000,2))

kable(inc.ex.study,

caption = "Incidence by main exposure status by study")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 13**. Incidence by main exposure status by study.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **main.exposure** | **studycode** | **py.ex.study** | **cancer** | **cancer.inc** |
| No | study1 | 258,159 | 103 | 39.9 |
| No | study2 | 408,875 | 137 | 33.5 |
| No | study3 | 257,285 | 267 | 103.8 |
| No | study4 | 335,294 | 118 | 35.2 |
| No | study5 | 982,283 | 419 | 42.7 |
| No | study6 | 225,515 | 95 | 42.1 |
| No | study7 | 515,011 | 374 | 72.6 |
| No | study8 | 846,288 | 581 | 68.7 |
| No | study9 | 810,505 | 429 | 52.9 |
| Yes | study1 | 11,032 | 19 | 172.2 |
| Yes | study2 | 17,312 | 18 | 104.0 |
| Yes | study3 | 10,545 | 14 | 132.8 |
| Yes | study4 | 21,271 | 27 | 126.9 |
| Yes | study5 | 38,114 | 52 | 136.4 |
| Yes | study6 | 12,698 | 21 | 165.4 |
| Yes | study7 | 33,481 | 53 | 158.3 |
| Yes | study8 | 54,988 | 77 | 140.0 |
| Yes | study9 | 67,023 | 92 | 137.3 |

Source: Table by author(s).

### Kaplan-Meier Plot of Time to Cancer

The example below creates Kaplan-Meier plot by “main.exposure” status for the pooled data using *ggsurvplot()* of the package “*survminer” [44]*: *Surv(ttcancer, cancer)* creates the survival object, where *ttcancer* is the time and *cancer* is the event indicator (1=event, 0=censored); *~ main.exposure*: asks *survfit* to calculate separate curves for the "Yes" and "No" exposure groups; *ggsurvplot(...)* of the *survminer* package creates a publication-ready plot of the *survfit* object.

candat$cancer<-as.numeric(as.character(candat$cancer)) candat$main.exposure<-as.factor(candat$main.exposure)

*# Fit the model*

fit<-survfit(Surv(ttcancer,cancer)~main.exposure, data = candat)

*# Plot survfit object*

ggsurvplot(fit, linetype = "strata",

conf.int = TRUE,pval = TRUE,censor = FALSE, palette = "redblue",

risk.table = T, cumevents = T, tables.height = 0.2,

break.x.by = 5,

ylim = c(0.7, 1),

xlim = c(0,40),

font.x = 6, font.y = 6,

font.legend = 6,

font.tickslab = 6,

pval.coord = c(1,0.8),

xlab = "Follow-up year",

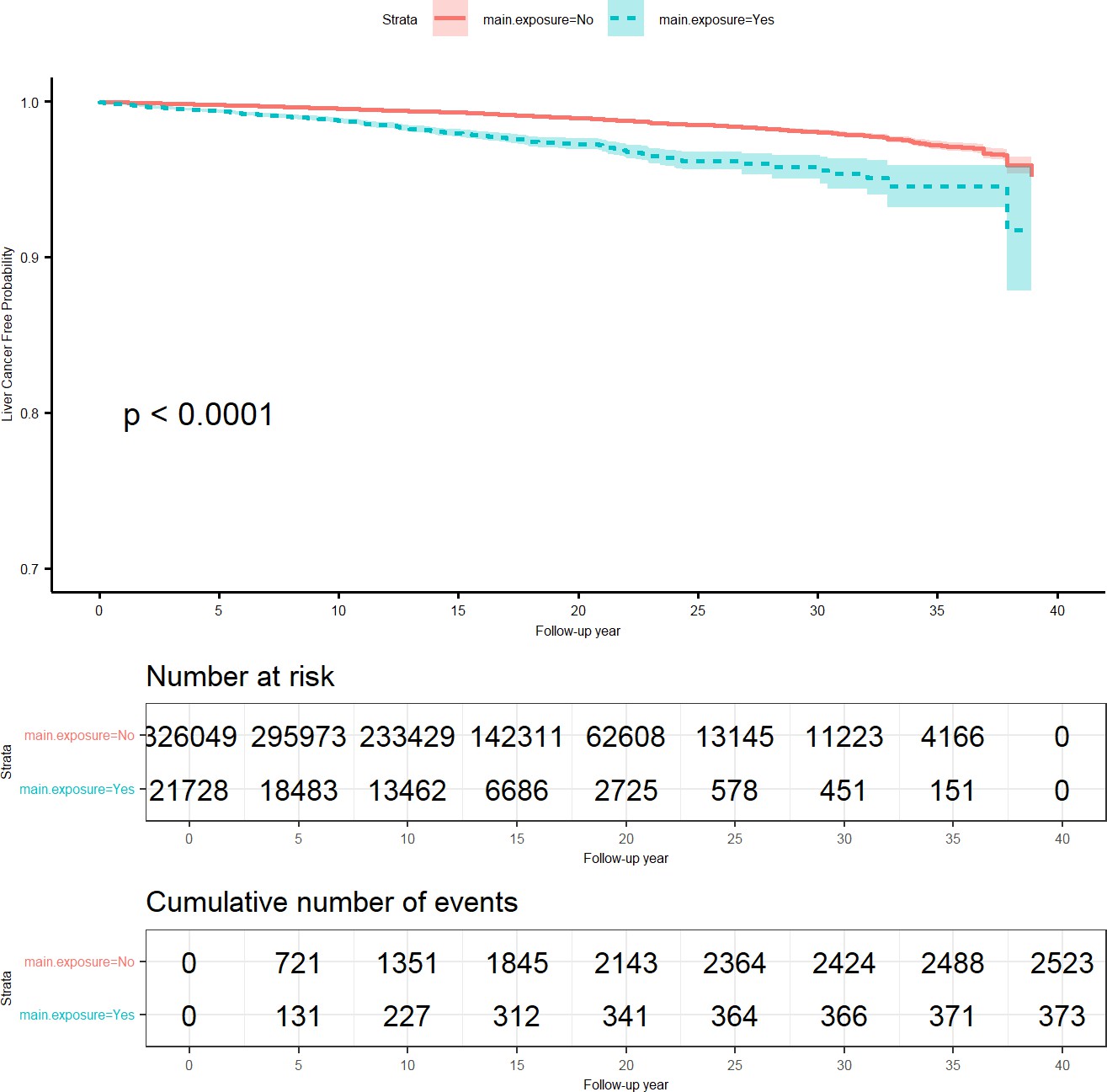
ylab = "Liver Cancer Free Probability", tables.theme = theme\_bw() +

theme(

axis.text = element\_text(size = 6), *# smaller axis text* axis.title = element\_text(size = 6), *# smaller axis titles* strip.text = element\_text(size = 6), *# smaller facet/strip text* legend.text = element\_text(size = 6)

)

)



Source: Figure by author(s).

**Figure 87**. Kaplan-Meier curve overall pooled data.

The KM plot shows that the "main.exposure=Yes" group (blue line) has a lower cancer-free probability over time compared to the "No" group (red line). The p < 0.0001 (from a log-rank test) confirms this difference in the pooled data is highly statistically significant.

### Cox Models

#### Unadjusted Cox Model

First, we run a simple (unadjusted) Cox model on the pooled data.

cfit<-coxph(Surv(ttcancer,cancer)~main.exposure, data = candat)

summary(cfit)

## Call: coxph(formula = Surv(ttcancer, cancer) ~ main.exposure, data =

## candat) n = 347777, number of events = 2896 (27033 observations

## deleted due to missingness)

## coef exp(coef) se(coef) z

## main.exposureYes 0.9784 2.6601 0.0556 17.6 Pr(>|z|)

## main.exposureYes <0.0000000000000002 \*\*\* — Signif. codes: 0 ‘***’ 0.001 ’****’*

## *0.01 ’*’ 0.05 ‘.’ 0.1 ’ ’ 1

## exp(coef) exp(-coef) lower .95 upper .95

## main.exposureYes 2.66 0.376 2.39 2.97

## Concordance = 0.543 (se = 0.004 ) Likelihood ratio test = 244 on 1

## df, p = <0.0000000000000002 Wald test = 310 on 1 df,

## p = <0.0000000000000002 Score

## (logrank) test = 336 on 1 df, p = <0.0000000000000002

rtab<-c(summary(cfit)$coefficients,

summary(cfit)$conf.int[,2:4],

summary(cfit)$logtest[3])

rtab<-ifelse(rtab> = 0.0001,round(rtab,4),"<0.0001")

rtab<-c(rownames(summary(cfit)$coefficients),rtab)

names(rtab)<-c("com.grp",colnames(summary(cfit)$coefficients),

names(summary(cfit)$conf.int[,2:4]),

"lr.p")

kable(rtab, caption = "Summary of unadjusted Cox model")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 14**. Summary of unadjusted Cox model.

x

com.grp main.exposureYes coef 0.9784

exp(coef) 2.6601

se(coef) 0.0556

z 17.6099

Pr(>|z|) <0.0001

exp(-coef) 0.3759

lower .95 2.3857

upper .95 2.9661

lr.p <0.0001

Source: Table by author(s).

The output shows that the Hazard Ratio (HR) is 2.6617. This is interpreted as: "In the pooled, unadjusted analysis, the exposed group has 2.66 times the hazard (or instantaneous risk) of developing cancer at any given time point compared to the unexposed group."

#### Adjusted Cox Model

Now we fit a Cox model adjusted for covariates (adding covariates to the above Cox model) and stratified by study (*+ strata(studycode)*). *+ strata(studycode)* tells the Cox model to assume a different, unique baseline hazard for each study. It does not estimate an effect for “*studycode”*. This is a method for accounting for heterogeneity between studies.

cfit<-coxph(Surv(ttcancer,cancer)~main.exposure +

gender + age + bmi +

smoking + alcoholuse +

strata(studycode), data = candat)

rtab<-cbind(summary(cfit)$coefficients,

summary(cfit)$conf.int[,2:4])

rtab<-ifelse(rtab> = 0.0001,round(rtab,4),"<0.0001")

rtab<-cbind(rownames(summary(cfit)$coefficients),rtab)

rownames(rtab)<-NULL

kable(rtab, caption = "Summary of adjusted Cox model")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 15**. Summary of adjusted Cox model.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **coef** | **exp(coef)** | **se(coef)** | **z** | **Pr(>|z|)** | **exp(-coef)** | **lower .95** | **upper .95** |
| main.exposureYes | 0.7502 | 2.1175 | 0.0596 | 12.5936 | <0.0001 | 0.4723 | 1.8842 | 2.3798 |
| genderMale | 0.835 | 2.3049 | 0.0598 | 13.9747 | <0.0001 | 0.4339 | 2.0502 | 2.5913 |
| age | 0.0556 | 1.0572 | 0.0021 | 25.9292 | <0.0001 | 0.9459 | 1.0528 | 1.0617 |
| bmi | <0.0001 | 1 | <0.0001 | 0.0727 | 0.942 | 1 | 0.9999 | 1.0001 |
| smokingYes | 0.4785 | 1.6136 | 0.0551 | 8.6845 | <0.0001 | 0.6197 | 1.4485 | 1.7976 |
| alcoholuseYes | 0.0975 | 1.1024 | 0.0479 | 2.0342 | 0.0419 | 0.9071 | 1.0036 | 1.211 |

Source: Table by author(s).

The output shows that after adjusting for confounders and stratifying by study, the HR for “main.exposureYes” is 2.1175.

### Subgroup Analysis

#### New Function for Subgroup Analysis, Heterogeneity Test

For a complex analysis and result summary that will be done multiple times, writing a new function for that analysis would be more efficient. Below is an example function *subhe()* written to perform and summarize neatly the results of Cox model for subgroups, heterogeneity test, Cox model unadjusted or adjusted for the subgroup variable with and without having “studycode” as strata in the model. With this function *subhe(),* one just need to specify the variables corresponding to time to event (“*ttevent*”), event (“*event*”), comparison variable (“*comvar*”), variables to be adjusted (“*adjustvar*”), subgroup variable (“*subvar*”), and the dataset to be used (“*data*”) and the function will do all the computation and summarize the results. The results of the following will be summarized and displayed as rows:

* The Cox model results comparing between the groups of the comparison variable (*comvar*) for each stratum of the subgroup variable (*subvar*). For example, for *comvar= “main.exposure”* and *subvar= “gender”*, the univariable Cox model results comparing between “main.exposureYes” (vs. “main.exposureNo”) will be displayed for each subgroup (stratum) “Female” and “Male”(rows “gender.stratified Female main.exposureYes” and “gender.stratified Male main.exposureYes”).
* The results of the adjusted Cox model including the comparison variable (*comvar*) and the subgroup variable (*subvar*) for each of these two variables. For example, for *comvar= “main.exposure”* and *subvar= “gender”*, the adjusted Cox model includes “main.exposure” and “gender”, and the results for each of these two variables will be displayed (rows “main.exposure.gender.adjusted main.exposureYes” and “main.exposure.gender.adjusted genderMale”).
* The results of the adjusted Cox model including the comparison variable (*comvar*), the subgroup variable (*subvar*) and *strata(studycode)*. For example, for *comvar= “main.exposure”* and *subvar= “gender”*, the adjusted Cox model includes “main.exposure”, “gender” and *strata(studycode)*, and the results for “main.exposure” and “gender” will be displayed (rows “main.exposure.gender.adjusted.strata main.exposureYes” and “main.exposure.gender.adjusted.strata genderMale”).
* The results of the unadjusted (univariable) Cox model including the comparison variable (*comvar*) and the results of the Cox model including the comparison variable (*comvar*) and *strata(studycode)* using the same dataset with non-missing values as the adjusted Cox models above. For example, for *comvar= “main.exposure”* and *subvar= “gender”*, the unadjusted Cox model comparing between “main.exposure” groups without and with *strata(studycode)* using the same dataset with non-missing values for “main.exposure” and “gender” will be displayed in result (rows “main.exposure.gender.unadjusted main.exposureYes” and “main.exposure.gender.unadjusted.strata main.exposureYes”).

The output table columns display the number of studies with available data included in the analysis (“number.study”), the number of subjects included in the analysis (“number.subject”), Cox model summarized results such as HR (“exp(coef)”) and 95%CI (“lower.95”, “upper.95”), p-value from likelihood ratio test for the significance of a variable (“lr.p”), and p-value for heterogeneity (“p.heterogeneity”) (interaction term between the comparison variable (*comvar*) and the subgroup variable(*subvar*)).

subhe<-function(ttevent,event,comvar,adjustvar,subvar,data){ require(survival)

data[,subvar]<-as.factor(data[,subvar]) fit<-list()

fit.re<-NULL

for (i in 1:nlevels(data[,subvar])){

datsub<-subset(data,data[,subvar] %in% levels(data[,subvar])[i]) nstudy<-length(unique(datsub[,"studycode"][!is.na(datsub[,subvar])])) nsubj<-nrow(na.omit(datsub[,c(comvar,ttevent,event)]))

fit[[i]]<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), comvar,sep = "~")),

data = datsub)

rtab<-c(paste(subvar,"stratified",sep = "."),levels(data[,subvar])[i],

rownames(summary(fit[[i]])$coefficients), nstudy,nsubj, round(summary(fit[[i]])$coefficients,4), round(summary(fit[[i]])$conf.int[,2:4],4),

round(summary(fit[[i]])$logtest[3],6)) names(rtab)<-c("varname","subgroup","comvar",

"number.study","number.subject", colnames(summary(fit[[i]])$coefficients), names(summary(fit[[i]])$conf.int[,2:4]), "lr.p")

fit.re<-rbind(fit.re,rtab)

}

*#full dataset #heterogeneity test*

f<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",sep = ""),

paste(comvar,subvar,sep = "+"),sep = "~")),

data = data)

fs<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), paste(comvar,subvar,

paste(comvar,subvar, sep = "\*"),

sep = "+"), sep = "~")),

data = data)

ano<-anova(f,fs)

p.he<-round(ano$`Pr(>|Chi|)`[2],6)

fit.he<-cbind(fit.re,p.heterogeneity = p.he)

*#adjusted for subgroup var*

datad<-na.omit(data[,c("studycode",comvar,

subvar,ttevent,event)]) nstudy<-length(unique(datad[,"studycode"])) nsubj<-nrow(datad)

fa<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), paste(comvar,subvar,sep = "+"), sep = "~")),

data = data)

atab<-cbind(paste(comvar,subvar,"adjusted",sep = "."),"", rownames(summary(fa)$coefficients), nstudy,nsubj, round(summary(fa)$coefficients,4), round(summary(fa)$conf.int[,2:4],4), round(summary(fa)$logtest[3],6))

colnames(atab)<-c("varname","subgroup","comvar",

"number.study","number.subject", colnames(summary(fa)$coefficients), colnames(summary(fa)$conf.int[,2:4]), "lr.p")

*#adjusted and stratified by study*

fas<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), paste(comvar,subvar,"strata(studycode)",

sep = "+"),

sep = "~")),

data = data)

astab<-cbind(paste(comvar,subvar,"adjusted.strata",sep = "."),"", rownames(summary(fas)$coefficients),

nstudy,nsubj, round(summary(fas)$coefficients,4), round(summary(fas)$conf.int[,2:4],4), round(summary(fas)$logtest[3],6))

colnames(astab)<-c("varname","subgroup","comvar",

"number.study","number.subject", colnames(summary(fas)$coefficients), colnames(summary(fas)$conf.int[,2:4]), "lr.p")

*# add unadjusted analysis with the same # non-missing sample size with adjusted*

fu<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), comvar,sep = "~")),

data = datad)

utab<-c(paste(comvar,subvar,"unadjusted",sep = "."),"", rownames(summary(fu)$coefficients), nstudy,nsubj, as.vector(round(summary(fu)$coefficients,4)), as.vector(round(summary(fu)$conf.int[,2:4],4)), as.vector(round(summary(fu)$logtest[3],6)))

names(utab)<-c("varname","subgroup","comvar",

"number.study","number.subject", colnames(summary(fu)$coefficients), names(summary(fu)$conf.int[,2:4]), "lr.p")

*#unadjusted but stratified by study*

fus<-coxph(as.formula(paste(paste("Surv(",ttevent,",",event,")",

sep = ""), paste(comvar,"strata(studycode)",

sep = "+"),

sep = "~")),

data = datad)

ustab<-c(paste(comvar,subvar,"unadjusted.strata",sep = "."),"", rownames(summary(fus)$coefficients),

nstudy,nsubj, as.vector(round(summary(fus)$coefficients,4)), as.vector(round(summary(fus)$conf.int[,2:4],4)), as.vector(round(summary(fus)$logtest[3],6)))

names(ustab)<-c("varname","subgroup","comvar",

"number.study","number.subject", colnames(summary(fus)$coefficients), names(summary(fus)$conf.int[,2:4]), "lr.p")

*#combine results*

austab<-rbind(atab,astab,utab,ustab) austab<-as.data.frame(austab) fit.he<-as.data.frame(fit.he)

fit.all<-plyr::rbind.fill(fit.he,austab)

fit.all

}

Now, we apply the above function *subhe()* for multiple subgroup variables ("gender","age.cat","bmi.cat","smoking","alcoholuse") by writing a loop. The loop will perform the *subhe()* function for each of the subgroup variable and add the results to a table. The output will be a big table combining the results for each of all variables.

subvec<-c("gender","age.cat","bmi.cat","smoking","alcoholuse")

candat[,subvec]<-lapply(candat[,subvec], as.factor)

tabsub<-NULL

for (i in 1:length(subvec)){

tabv<-subhe(ttevent = "ttcancer",

event = "cancer",

comvar = "main.exposure",

subvar = subvec[i], data = candat)

tabsub<-rbind(tabsub,tabv)

}

kable(tabsub, caption = "Summary of subgroup analysis results")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 16**. Summary of subgroup analysis results.

187

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **varname** | **subgroup** | **comvar** | **number.study** | **number.subject** | **coef** | **exp(coef)** | **se(coef)** | **z** | **Pr(>|z|)** | **exp(-coef)** | **lower .95** | **upper .95** | **lr.p** | **p.heterogeneity** |
| gender.stratified | Female | main.exposureYes | 9 | 186,759 | 0.9527 | 2.5927 | 0.11 | 8.6644 | 0 | 0.3857 | 2.0901 | 3.2162 | 0 | 0.564023 |
| gender.stratified | Male | main.exposureYes | 9 | 161,018 | 0.8742 | 2.3969 | 0.0644 | 13.5643 | 0 | 0.4172 | 2.1125 | 2.7196 | 0 | 0.564023 |
| main.exposure.gender.adjusted |  | main.exposureYes | 9 | 347,777 | 0.8954 | 2.4484 | 0.0556 | 16.0982 | 0 | 0.4084 | 2.1955 | 2.7304 | 0 | NA |
| main.exposure.gender.adjusted |  | genderMale | 9 | 347,777 | 1.0901 | 2.9747 | 0.0409 | 26.657 | 0 | 0.3362 | 2.7455 | 3.2229 | 0 | NA |
| main.exposure.gender.adjusted.strata |  | main.exposureYes | 9 | 347,777 | 0.8888 | 2.4322 | 0.0558 | 15.9387 | 0 | 0.4111 | 2.1804 | 2.7131 | 0 | NA |
| main.exposure.gender.adjusted.strata |  | genderMale | 9 | 347,777 | 1.1192 | 3.0625 | 0.041 | 27.3137 | 0 | 0.3265 | 2.8262 | 3.3186 | 0 | NA |
| main.exposure.gender.unadjusted |  | main.exposureYes | 9 | 347,777 | 0.9784 | 2.6601 | 0.0556 | 17.6099 | 0 | 0.3759 | 2.3857 | 2.9661 | 0 | NA |
| main.exposure.gender.unadjusted.strata |  | main.exposureYes | 9 | 347,777 | 0.9621 | 2.6173 | 0.0558 | 17.2573 | 0 | 0.3821 | 2.3463 | 2.9195 | 0 | NA |
| age.cat.stratified | <40 years old | main.exposureYes | 4 | 19,151 | 0.2324 | 1.2617 | 0.5104 | 0.4553 | 0.6489 | 0.7926 | 0.4639 | 3.4311 | 0.660216 | 0.008634 |
| age.cat.stratified | > = 40&<60 years old | main.exposureYes | 9 | 208,460 | 1.0488 | 2.8541 | 0.0827 | 12.6844 | 0 | 0.3504 | 2.4271 | 3.3562 | 0 | 0.008634 |
| age.cat.stratified | > = 60 years old | main.exposureYes | 9 | 120,166 | 0.6929 | 1.9995 | 0.0761 | 9.1055 | 0 | 0.5001 | 1.7224 | 2.3211 | 0 | 0.008634 |
| main.exposure.age.cat.adjusted |  | main.exposureYes | 9 | 347,777 | 0.8494 | 2.3383 | 0.0558 | 15.2184 | 0 | 0.4277 | 2.096 | 2.6086 | 0 | NA |
| main.exposure.age.cat.adjusted |  | age.cat> = 40&<60 years old | 9 | 347,777 | 0.8253 | 2.2826 | 0.1054 | 7.8329 | 0 | 0.4381 | 1.8567 | 2.8062 | 0 | NA |
| main.exposure.age.cat.adjusted |  | age.cat> = 60 years old | 9 | 347,777 | 1.6624 | 5.272 | 0.1073 | 15.4946 | 0 | 0.1897 | 4.2722 | 6.5058 | 0 | NA |
| main.exposure.age.cat.adjusted.strata |  | main.exposureYes | 9 | 347,777 | 0.8349 | 2.3046 | 0.056 | 14.9142 | 0 | 0.4339 | 2.0651 | 2.5719 | 0 | NA |
| main.exposure.age.cat.adjusted.strata |  | age.cat> = 40&<60 years old | 9 | 347,777 | 1.0183 | 2.7684 | 0.1064 | 9.5694 | 0 | 0.3612 | 2.2473 | 3.4104 | 0 | NA |
| main.exposure.age.cat.adjusted.strata |  | age.cat> = 60 years old | 9 | 347,777 | 1.7948 | 6.0184 | 0.1088 | 16.504 | 0 | 0.1662 | 4.8631 | 7.4482 | 0 | NA |
| main.exposure.age.cat.unadjusted |  | main.exposureYes | 9 | 347,777 | 0.9784 | 2.6601 | 0.0556 | 17.6099 | 0 | 0.3759 | 2.3857 | 2.9661 | 0 | NA |
| main.exposure.age.cat.unadjusted.strata |  | main.exposureYes | 9 | 347,777 | 0.9621 | 2.6173 | 0.0558 | 17.2573 | 0 | 0.3821 | 2.3463 | 2.9195 | 0 | NA |
| bmi.cat.stratified | <23 | main.exposureYes | 8 | 143,748 | 0.9911 | 2.6942 | 0.1053 | 9.4113 | 0 | 0.3712 | 2.1918 | 3.3119 | 0 | 0.725015 |
| bmi.cat.stratified | > = 23 | main.exposureYes | 8 | 150,120 | 1.0468 | 2.8484 | 0.0804 | 13.0211 | 0 | 0.3511 | 2.4332 | 3.3345 | 0 | 0.725015 |
| main.exposure.bmi.cat.adjusted |  | main.exposureYes | 8 | 293,868 | 1.0257 | 2.789 | 0.0639 | 16.0621 | 0 | 0.3586 | 2.4609 | 3.1608 | 0 | NA |
| main.exposure.bmi.cat.adjusted |  | bmi.cat> = 23 | 8 | 293,868 | 0.0761 | 1.079 | 0.0438 | 1.7377 | 0.0823 | 0.9268 | 0.9903 | 1.1757 | 0 | NA |
| main.exposure.bmi.cat.adjusted.strata |  | main.exposureYes | 8 | 293,868 | 1.0117 | 2.7504 | 0.0641 | 15.7794 | 0 | 0.3636 | 2.4256 | 3.1187 | 0 | NA |
| main.exposure.bmi.cat.adjusted.strata |  | bmi.cat> = 23 | 8 | 293,868 | 0.0555 | 1.0571 | 0.0441 | 1.2582 | 0.2083 | 0.946 | 0.9695 | 1.1526 | 0 | NA |
| main.exposure.bmi.cat.unadjusted |  | main.exposureYes | 8 | 293,868 | 1.0338 | 2.8118 | 0.0637 | 16.2335 | 0 | 0.3556 | 2.4818 | 3.1856 | 0 | NA |
| main.exposure.bmi.cat.unadjusted.strata |  | main.exposureYes | 8 | 293,868 | 1.0175 | 2.7664 | 0.064 | 15.9113 | 0 | 0.3615 | 2.4405 | 3.1358 | 0 | NA |
| smoking.stratified | No | main.exposureYes | 9 | 178,259 | 1.0493 | 2.8558 | 0.1007 | 10.4248 | 0 | 0.3502 | 2.3445 | 3.4786 | 0 | 0.117492 |
| smoking.stratified | Yes | main.exposureYes | 9 | 139,527 | 0.8577 | 2.3578 | 0.069 | 12.4224 | 0 | 0.4241 | 2.0593 | 2.6994 | 0 | 0.117492 |
| main.exposure.smoking.adjusted |  | main.exposureYes | 9 | 317,786 | 0.917 | 2.5017 | 0.057 | 16.0914 | 0 | 0.3997 | 2.2373 | 2.7973 | 0 | NA |
| main.exposure.smoking.adjusted |  | smokingYes | 9 | 317,786 | 0.9709 | 2.6402 | 0.0407 | 23.8569 | 0 | 0.3788 | 2.4378 | 2.8594 | 0 | NA |
| main.exposure.smoking.adjusted.strata |  | main.exposureYes | 9 | 317,786 | 0.9006 | 2.4611 | 0.0572 | 15.7537 | 0 | 0.4063 | 2.2002 | 2.7528 | 0 | NA |
| main.exposure.smoking.adjusted.strata |  | smokingYes | 9 | 317,786 | 1.041 | 2.832 | 0.0413 | 25.2072 | 0 | 0.3531 | 2.6118 | 3.0708 | 0 | NA |
| main.exposure.smoking.unadjusted |  | main.exposureYes | 9 | 317,786 | 0.9824 | 2.6708 | 0.0569 | 17.2544 | 0 | 0.3744 | 2.3888 | 2.9861 | 0 | NA |
| main.exposure.smoking.unadjusted.strata |  | main.exposureYes | 9 | 317,786 | 0.9697 | 2.637 | 0.0571 | 16.9704 | 0 | 0.3792 | 2.3577 | 2.9495 | 0 | NA |
| alcoholuse.stratified | No | main.exposureYes | 8 | 151,202 | 0.984 | 2.6751 | 0.0868 | 11.332 | 0 | 0.3738 | 2.2565 | 3.1714 | 0 | 0.741064 |
| alcoholuse.stratified | Yes | main.exposureYes | 8 | 143,403 | 0.9479 | 2.5803 | 0.0779 | 12.1674 | 0 | 0.3875 | 2.2149 | 3.006 | 0 | 0.741064 |
| main.exposure.alcoholuse.adjusted |  | main.exposureYes | 8 | 294,605 | 0.9653 | 2.6255 | 0.058 | 16.6495 | 0 | 0.3809 | 2.3435 | 2.9414 | 0 | NA |
| main.exposure.alcoholuse.adjusted |  | alcoholuseYes | 8 | 294,605 | 0.3914 | 1.479 | 0.0394 | 9.9437 | 0 | 0.6761 | 1.3692 | 1.5976 | 0 | NA |
| main.exposure.alcoholuse.adjusted.strata |  | main.exposureYes | 8 | 294,605 | 0.9502 | 2.5862 | 0.0582 | 16.3378 | 0 | 0.3867 | 2.3076 | 2.8985 | 0 | NA |
| main.exposure.alcoholuse.adjusted.strata |  | alcoholuseYes | 8 | 294,605 | 0.4861 | 1.6259 | 0.0418 | 11.6388 | 0 | 0.615 | 1.4981 | 1.7646 | 0 | NA |
| main.exposure.alcoholuse.unadjusted |  | main.exposureYes | 8 | 294,605 | 0.9621 | 2.6171 | 0.058 | 16.5919 | 0 | 0.3821 | 2.3359 | 2.9321 | 0 | NA |
| main.exposure.alcoholuse.unadjusted.strata |  | main.exposureYes | 8 | 294,605 | 0.9564 | 2.6023 | 0.0582 | 16.4374 | 0 | 0.3843 | 2.3218 | 2.9167 | 0 | NA |

Source: Table by author(s).

### Analysis of Each Study in the Consortium

#### Kaplan-Meir Curve and Univariable Cox Model for Each Study

We can write a loop to perform the same analysis (Kaplan-Meier curve and univariable Cox model) for each study. Then we combine all the plots into a multi-panel plot (using *wrap\_plots()*) and combine the results of univariable Cox model of each of all studies in a table:

cname<-unique(candat$studycode[!is.na(candat$main.exposure)]) fitl<-list()

splotl<-list() cfitl<-list() rtabl<-list() cfit.re<-NULL

for (i in 1:length(cname)){

*# Survival curves*

fitl[[i]]<-survfit(Surv(ttcancer,cancer)~main.exposure,

data = subset(candat,studycode = = cname[i])) sp<-ggsurvplot(fitl[[i]], linetype = "strata",

conf.int = TRUE,pval = TRUE,censor = FALSE, palette = "grey", ylim = c(0.7, 1), pval.coord = c(1,0.8),

xlab = "Follow-up year",

ylab = "Cancer Free Probability", title = paste(cname[i]))

splotl[[i]]<-sp$plot

*# cox model*

cfitl[[i]]<-coxph(Surv(ttcancer,cancer)~main.exposure, data = subset(candat,studycode = = cname[i]))

rtabl[[i]]<-cbind(matrix(cname[i],ncol = 1),

matrix(rownames(summary(cfitl[[i]])$coefficients), ncol = 1),

matrix(prettyNum(summary(cfitl[[i]])$coefficients), ncol = 5),

matrix(prettyNum(summary(cfitl[[i]])$conf.int[,2:4]), ncol = 3),

matrix(prettyNum(summary(cfitl[[i]])$logtest[3]), ncol = 1))

cfit.re<-rbind(cfit.re,rtabl[[i]])

}

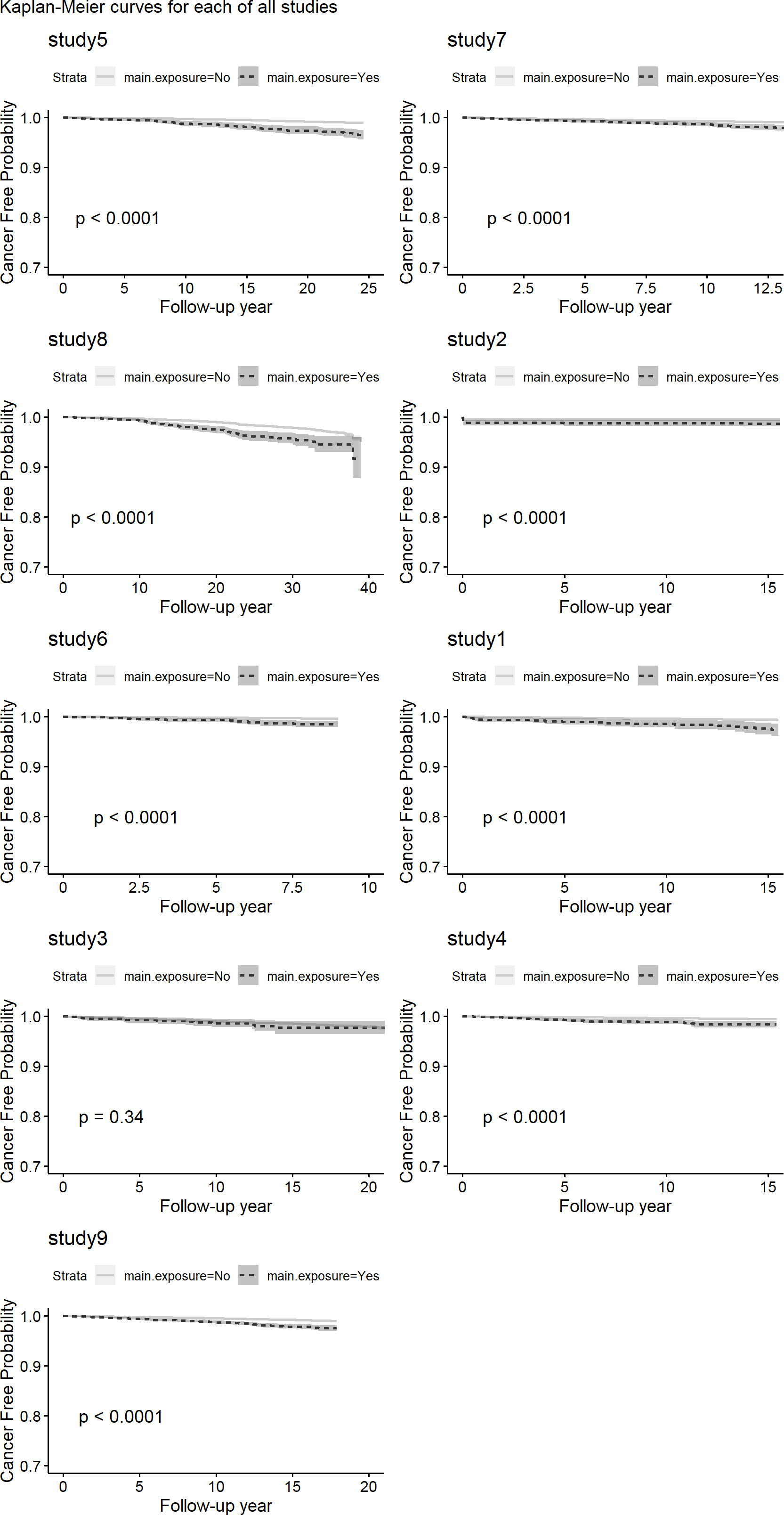
*# Combine all plots into a single figure (grid layout)*

combined\_plot <- wrap\_plots(splotl, ncol = 2) +

plot\_annotation(title = "Kaplan-Meier curves for each of all studies")

*# Print the combined figure*

combined\_plot



Source: Figure by author(s).

**Figure 88**. Kaplan-Meier curve for each of all studies.

Now we combine univariable Cox model results for each of all studies in a table.

colnames(cfit.re)<-c("study","com.grp",

"coef","hr","se.coef","z", "z.p",

"exp.minuscoef","ll.95.hr","ul.95.hr", "lr.p")

cfit.re<-as.data.frame(cfit.re)

*#calculate SE of HR*

cfit.re[,3:11]<-lapply(cfit.re[,3:11],as.character) cfit.re[,3:11]<-lapply(cfit.re[,3:11],as.numeric)

kable(cfit.re, caption = "Univariable Cox model results of each study")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 17**. Univariable Cox model results of each study.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **study** | **com.grp** | **coef** | **hr** | **se.coef** | **z** | **z.p** | **exp.minuscoef** | **ll.95.hr** | **ul.95.hr** | **lr.p** |
| study5 | main.exposureYes | 1.200 | 3.32 | 0.147 | 8.160 | 0.000 | 0.301 | 2.490 | 4.43 | 0.000 |
| study7 | main.exposureYes | 0.779 | 2.18 | 0.147 | 5.300 | 0.000 | 0.459 | 1.630 | 2.91 | 0.000 |
| study8 | main.exposureYes | 0.827 | 2.29 | 0.122 | 6.800 | 0.000 | 0.437 | 1.800 | 2.90 | 0.000 |
| study2 | main.exposureYes | 1.030 | 2.81 | 0.251 | 4.120 | 0.000 | 0.356 | 1.720 | 4.59 | 0.000 |
| study6 | main.exposureYes | 1.370 | 3.95 | 0.241 | 5.700 | 0.000 | 0.253 | 2.460 | 6.34 | 0.000 |
| study1 | main.exposureYes | 1.460 | 4.32 | 0.250 | 5.860 | 0.000 | 0.231 | 2.650 | 7.06 | 0.000 |
| study3 | main.exposureYes | 0.259 | 1.30 | 0.274 | 0.944 | 0.345 | 0.772 | 0.757 | 2.22 | 0.363 |
| study4 | main.exposureYes | 1.280 | 3.60 | 0.213 | 6.010 | 0.000 | 0.277 | 2.370 | 5.48 | 0.000 |
| study9 | main.exposureYes | 0.978 | 2.66 | 0.115 | 8.500 | 0.000 | 0.376 | 2.120 | 3.33 | 0.000 |

Source: Table by author(s).

#### Multivariable Cox Model for Each Study

Similar to the above, we can write a loop to perform the same multivariable Cox model for each study and combine results of multivariable Cox model of each of all studies in a table. Note that, we write *if* condition to perform Cox model only for the study with available data for all variables included in the multivariable model.

cfitl<-list() rtabl<-list() cfit.re.m<-NULL

for (i in 1:length(cname)){

*# cox model*

datas<-subset(candat,studycode = = cname[i]) if (nrow(na.omit(datas)) = = 0){

cfit.re.m<-cfit.re.m

}

if (nrow(na.omit(datas))>0){

cfitl[[i]]<-coxph(Surv(ttcancer,cancer)~main.exposure + gender + age + bmi +

smoking + alcoholuse, data = datas)

rtabl[[i]]<-cbind(matrix(cname[i],ncol = 1,

nrow = nrow(summary(cfitl[[i]])$coefficients)), matrix(rownames(summary(cfitl[[i]])$coefficients),

ncol = 1),

matrix(prettyNum(summary(cfitl[[i]])$coefficients), ncol = 5),

matrix(prettyNum(summary(cfitl[[i]])$conf.int[,2:4]), ncol = 3))

cfit.re.m<-rbind(cfit.re.m,rtabl[[i]])

}

}

colnames(cfit.re.m)<-c("study","com.grp",

"coef","hr","se.coef","z", "z.p",

"exp.minuscoef","ll.95.hr","ul.95.hr") cfit.re.m<-as.data.frame(cfit.re.m)

cfit.re.m[,3:10]<-lapply(cfit.re.m[,3:10],as.character)

cfit.re.m[,3:10]<-lapply(cfit.re.m[,3:10],as.numeric) kable(cfit.re.m,

caption = "Multivariable Cox model results of each study")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 18**. Multivariable Cox model results of each study.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **study** | **com.grp** | **coef** | **hr** | **se.coef** | **z** | **z.p** | **exp.minuscoef** | **ll.95.hr** | **ul.95.hr** |
| study5 | main.exposureYes | 0.840 | 2.320 | 0.162 | 5.190 | 0.000 | 0.432 | 1.690 | 3.18 |
| study5 | genderMale | 0.696 | 2.010 | 0.175 | 3.980 | 0.000 | 0.498 | 1.420 | 2.83 |
| study5 | age | 0.075 | 1.080 | 0.007 | 10.100 | 0.000 | 0.928 | 1.060 | 1.09 |
| study5 | bmi | 0.070 | 1.070 | 0.016 | 4.240 | 0.000 | 0.932 | 1.040 | 1.11 |
| study5 | smokingYes | 0.343 | 1.410 | 0.145 | 2.370 | 0.018 | 0.710 | 1.060 | 1.87 |
| study5 | alcoholuseYes | 0.540 | 1.720 | 0.148 | 3.640 | 0.000 | 0.583 | 1.280 | 2.29 |
| study7 | main.exposureYes | 0.646 | 1.910 | 0.157 | 4.130 | 0.000 | 0.524 | 1.400 | 2.59 |
| study7 | genderMale | 0.337 | 1.400 | 0.178 | 1.900 | 0.058 | 0.714 | 0.989 | 1.99 |
| study7 | age | 0.058 | 1.060 | 0.006 | 9.720 | 0.000 | 0.944 | 1.050 | 1.07 |
| study7 | bmi | 0.019 | 1.020 | 0.015 | 1.230 | 0.220 | 0.982 | 0.989 | 1.05 |
| study7 | smokingYes | 0.620 | 1.860 | 0.164 | 3.780 | 0.000 | 0.538 | 1.350 | 2.57 |
| study7 | alcoholuseYes | 0.161 | 1.170 | 0.143 | 1.120 | 0.262 | 0.852 | 0.887 | 1.55 |
| study6 | main.exposureYes | 1.020 | 2.770 | 0.307 | 3.310 | 0.001 | 0.362 | 1.510 | 5.05 |
| study6 | genderMale | 0.661 | 1.940 | 0.343 | 1.930 | 0.054 | 0.516 | 0.989 | 3.79 |
| study6 | age | 0.063 | 1.070 | 0.011 | 5.980 | 0.000 | 0.939 | 1.040 | 1.09 |
| study6 | bmi | −0.001 | 0.999 | 0.035 | −0.021 | 0.983 | 1.000 | 0.933 | 1.07 |
| study6 | smokingYes | 0.550 | 1.730 | 0.315 | 1.750 | 0.080 | 0.577 | 0.936 | 3.21 |
| study6 | alcoholuseYes | −0.137 | 0.872 | 0.286 | −0.477 | 0.634 | 1.150 | 0.498 | 1.53 |
| study1 | main.exposureYes | 1.180 | 3.260 | 0.255 | 4.630 | 0.000 | 0.306 | 1.980 | 5.39 |
| study1 | genderMale | 0.972 | 2.640 | 0.316 | 3.080 | 0.002 | 0.378 | 1.420 | 4.91 |
| study1 | age | 0.052 | 1.050 | 0.009 | 5.810 | 0.000 | 0.950 | 1.030 | 1.07 |
| study1 | bmi | 0.045 | 1.050 | 0.032 | 1.390 | 0.165 | 0.956 | 0.982 | 1.11 |
| study1 | smokingYes | 0.586 | 1.800 | 0.290 | 2.020 | 0.043 | 0.557 | 1.020 | 3.17 |
| study1 | alcoholuseYes | −0.094 | 0.911 | 0.242 | −0.387 | 0.698 | 1.100 | 0.567 | 1.46 |
| study3 | main.exposureYes | 0.112 | 1.120 | 0.276 | 0.407 | 0.684 | 0.894 | 0.651 | 1.92 |
| study3 | genderMale | 0.992 | 2.700 | 0.189 | 5.250 | 0.000 | 0.371 | 1.860 | 3.90 |
| study3 | age | 0.047 | 1.050 | 0.005 | 8.770 | 0.000 | 0.954 | 1.040 | 1.06 |

**Table 18.** *Cont.*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **study** | **com.grp** | **coef** | **hr** | **se.coef** | **z** | **z.p** | **exp.minuscoef** | **ll.95.hr** | **ul.95.hr** |
| study3 | bmi | −0.016 | 0.984 | 0.020 | −0.792 | 0.428 | 1.020 | 0.945 | 1.02 |
| study3 | smokingYes | 0.293 | 1.340 | 0.173 | 1.690 | 0.090 | 0.746 | 0.955 | 1.88 |
| study3 | alcoholuseYes | 0.319 | 1.380 | 0.148 | 2.160 | 0.031 | 0.727 | 1.030 | 1.84 |
| study4 | main.exposureYes | 0.608 | 1.840 | 0.235 | 2.580 | 0.010 | 0.545 | 1.160 | 2.91 |
| study4 | genderMale | 1.380 | 3.990 | 0.311 | 4.460 | 0.000 | 0.250 | 2.170 | 7.34 |
| study4 | age | 0.064 | 1.070 | 0.010 | 6.650 | 0.000 | 0.938 | 1.050 | 1.09 |
| study4 | bmi | 0.069 | 1.070 | 0.030 | 2.260 | 0.024 | 0.933 | 1.010 | 1.14 |
| study4 | smokingYes | 0.049 | 1.050 | 0.228 | 0.215 | 0.830 | 0.952 | 0.672 | 1.64 |
| study4 | alcoholuseYes | 0.149 | 1.160 | 0.248 | 0.600 | 0.548 | 0.862 | 0.714 | 1.89 |
| study9 | main.exposureYes | 0.751 | 2.120 | 0.117 | 6.440 | 0.000 | 0.472 | 1.690 | 2.66 |
| study9 | genderMale | 1.100 | 3.000 | 0.113 | 9.700 | 0.000 | 0.334 | 2.400 | 3.74 |
| study9 | age | 0.069 | 1.070 | 0.006 | 12.100 | 0.000 | 0.934 | 1.060 | 1.08 |
| study9 | bmi | 0.071 | 1.070 | 0.011 | 6.540 | 0.000 | 0.931 | 1.050 | 1.10 |
| study9 | smokingYes | 0.508 | 1.660 | 0.101 | 5.030 | 0.000 | 0.602 | 1.360 | 2.03 |
| study9 | alcoholuseYes | −0.096 | 0.909 | 0.110 | −0.869 | 0.385 | 1.100 | 0.733 | 1.13 |

Source: Table by author(s).

### Meta-Analysis Across Studies Based on the Results of Each Study

#### Univariable Cox Model

Now that we have the Cox model results of each of all studies from the above analysis. We can perform meta-analysis and forest plot of the results of univariable Cox model of all studies above using the “*meta”* package [66]: *metagen(TE = coef, seTE = se.coef, ...)* is meta-analysis function, *TE* stands for "Treatment Effect" (our log-HR) and *seTE* is its standard error, *sm = "HR"* specifies the summary measure is a Hazard Ratio, *forest(...)* generates the forest plot.

mcfit<-metagen(TE = coef,

seTE = se.coef,

studlab = study, sm = "HR",

data = cfit.re)

*# Save current margin settings*

old\_mar <- par("mar")

*# Increase margins to avoid cropping*

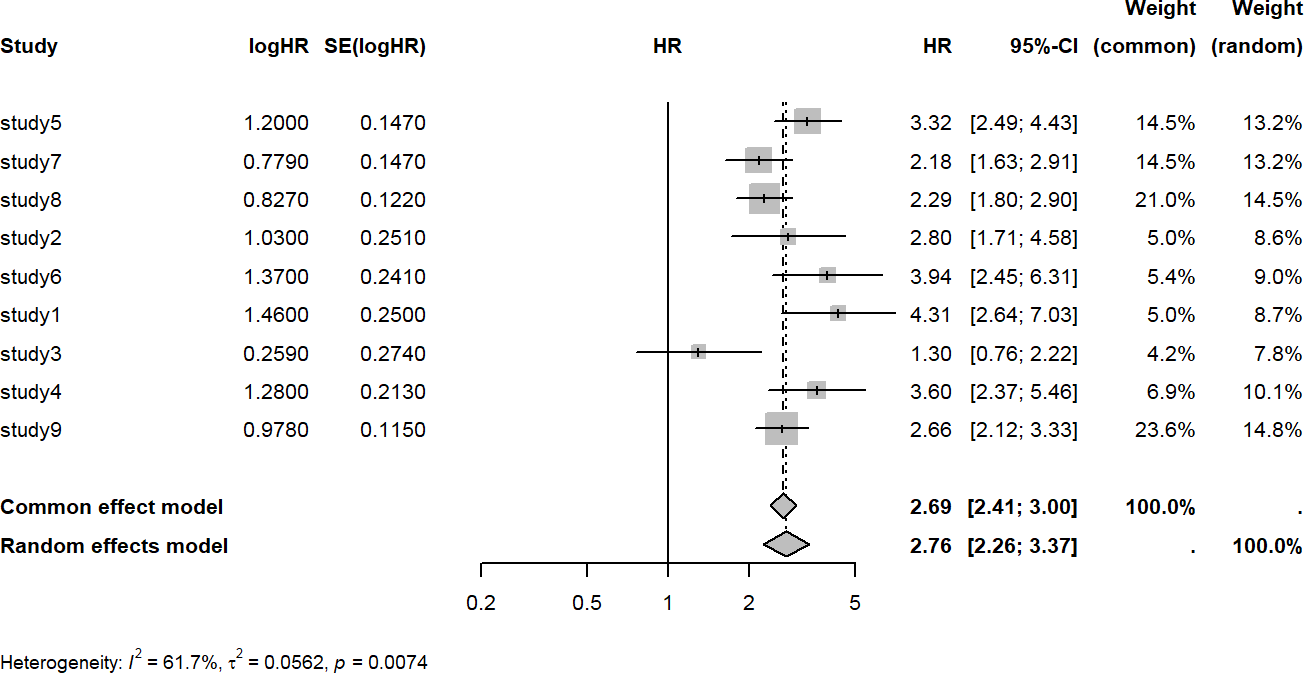
par(mar = c(5, 6, 4, 2))

forest(mcfit,

smlab = "HR", fontsize = 8)

*# Reset margins to original settings*

par(mar = old\_mar)



Source: Figure by author(s).

**Figure 89**. Metaanalysis univariable Cox model.

The output figure shows:

* The squares represent the HR from each individual study. The horizontal lines are their 95% CIs.
* The "Common effect model" (HR=2.69) is the pooled estimate if we assume the true effect is identical in all studies.
* The "Random effects model" (HR=2.76) is the pooled estimate allowing for the true effect to vary between studies.
* Heterogeneity: The statistics (I^2 = 61.7, p = 0.0074) show that there is significant heterogeneity. This means the random effects model is the more appropriate and realistic choice.

#### Multivariable Cox Model

We can perform similar meta-analysis and forest plot of the results of multivariable Cox model of all studies above (note that, we only do meta-analysis of the results of “main.exposure”):

mcfit.m<-metagen(TE = coef,

seTE = se.coef, studlab = study, sm = "HR",

data = subset(cfit.re.m,com.grp == "main.exposureYes"))

*# Save current margin settings*

old\_mar <- par("mar")

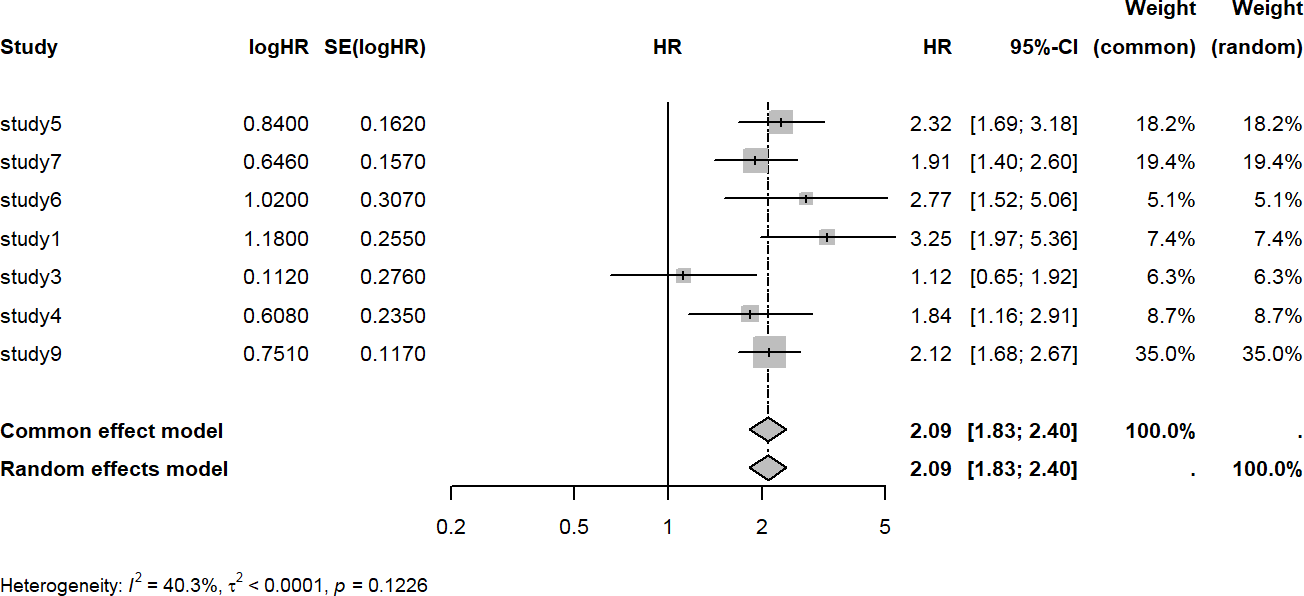
*# Increase margins to avoid cropping*

par(mar = c(5, 6, 4, 2))

forest(mcfit.m, smlab = "HR", fontsize = 8)

*# Reset margins to original settings*

par(mar = old\_mar)



Source: Figure by author(s).

**Figure 90**. Metaanalysis multivariable Cox model.

### Sensitivity Analysis (Leave One Study Out Validation)

The purpose of sensitivity analysis is to evaluate if any specific study strongly affects the meta-analysis results. The approach is to remove a study and re-perform the meta-analysis and repeat the procedure for all study (again, we can write a loop for this repeated analysis). Then, the meta-analysis results after removing each of the studies are combined and summarized in a plot. Below is an example of leave one study out validation for univariable Cox model. First, we remove a study and re-perform the meta-analysis and display the forest plot without the study which has been removed:

*# Save current margin settings*

old\_mar <- par("mar")

*# Increase margins to avoid cropping*

par(mar = c(5, 6, 4, 2))

cname<-unique(cfit.re$study)

datv<-list()

mcfitv<-list() hr.fixed<-NULL ll.fixed<-NULL ul.fixed<-NULL hr.random<-NULL ll.random<-NULL ul.random<-NULL

for (i in 1:length(cname)){

datv[[i]]<-cfit.re[!cfit.re$study %in% cname[i],]

mcfitv[[i]]<-metagen(TE = coef,

seTE = se.coef,

studlab = study,

sm = "HR",

data = datv[[i]])

*# Forest plot*

forest(mcfitv[[i]],smlab = paste("Uni","No",cname[i],sep = "\_"), fontsize = 8)

hr.fixed[i]<-exp(mcfitv[[i]]$TE.fixed)

ll.fixed[i]<-exp(mcfitv[[i]]$TE.fixed-1.96\*mcfitv[[i]]$seTE.fixed)

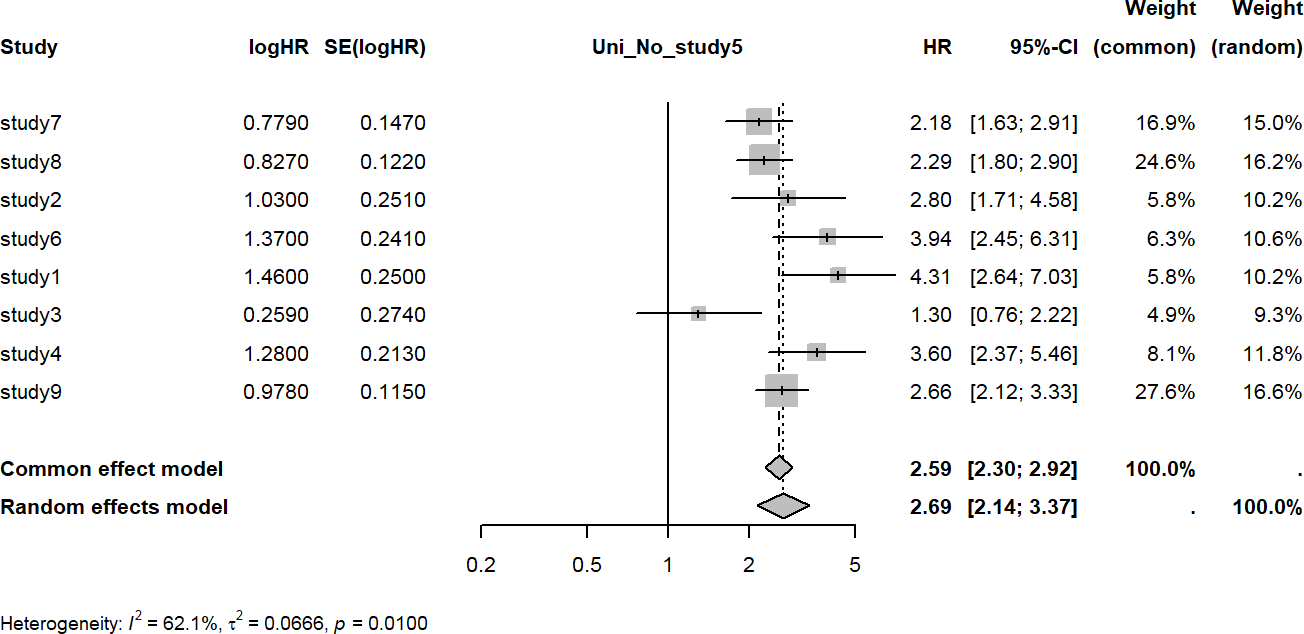
ul.fixed[i]<-exp(mcfitv[[i]]$TE.fixed+1.96\*mcfitv[[i]]$seTE.fixed)

hr.random[i]<-exp(mcfitv[[i]]$TE.random)

ll.random[i]<-exp(mcfitv[[i]]$TE.random-1.96\*mcfitv[[i]]$seTE.random)

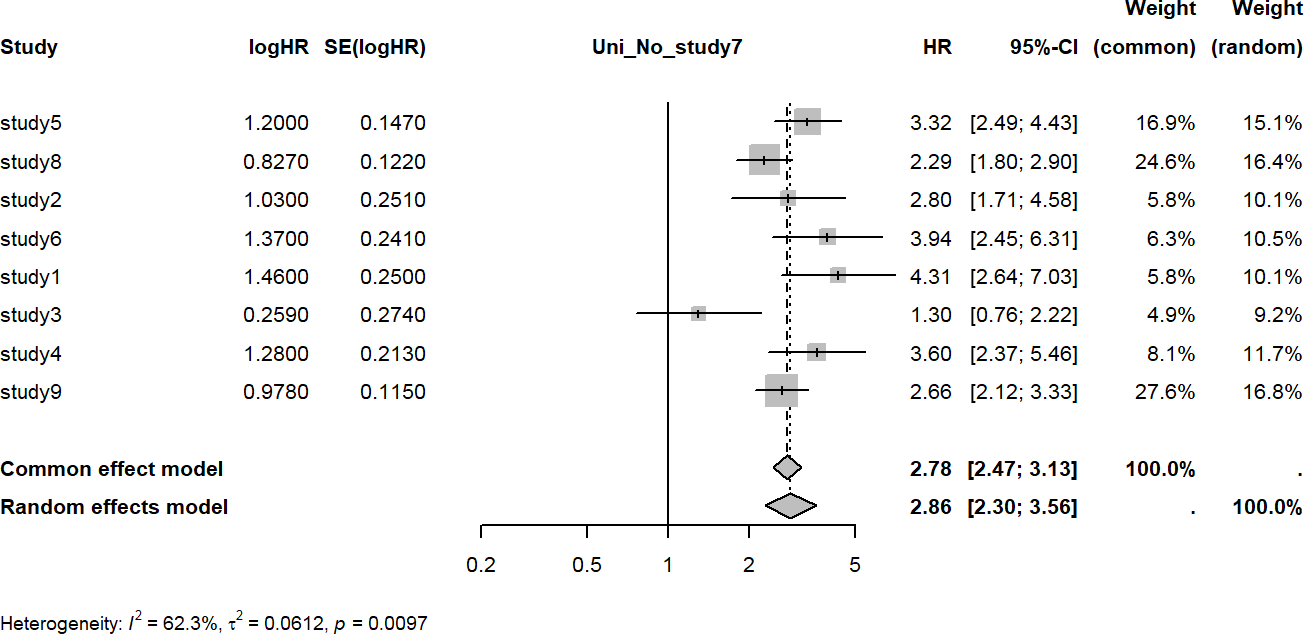
ul.random[i]<-exp(mcfitv[[i]]$TE.random+1.96\*mcfitv[[i]]$seTE.random)

}



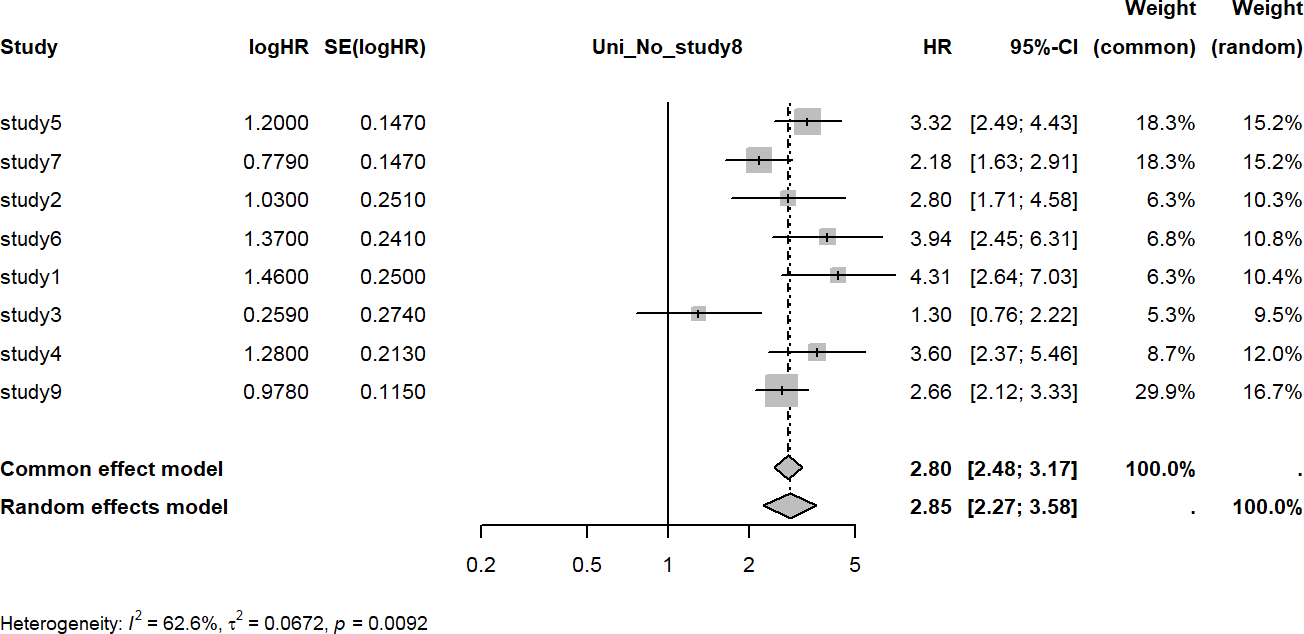
Source: Figure by author(s).

**Figure 91**. Study left out.



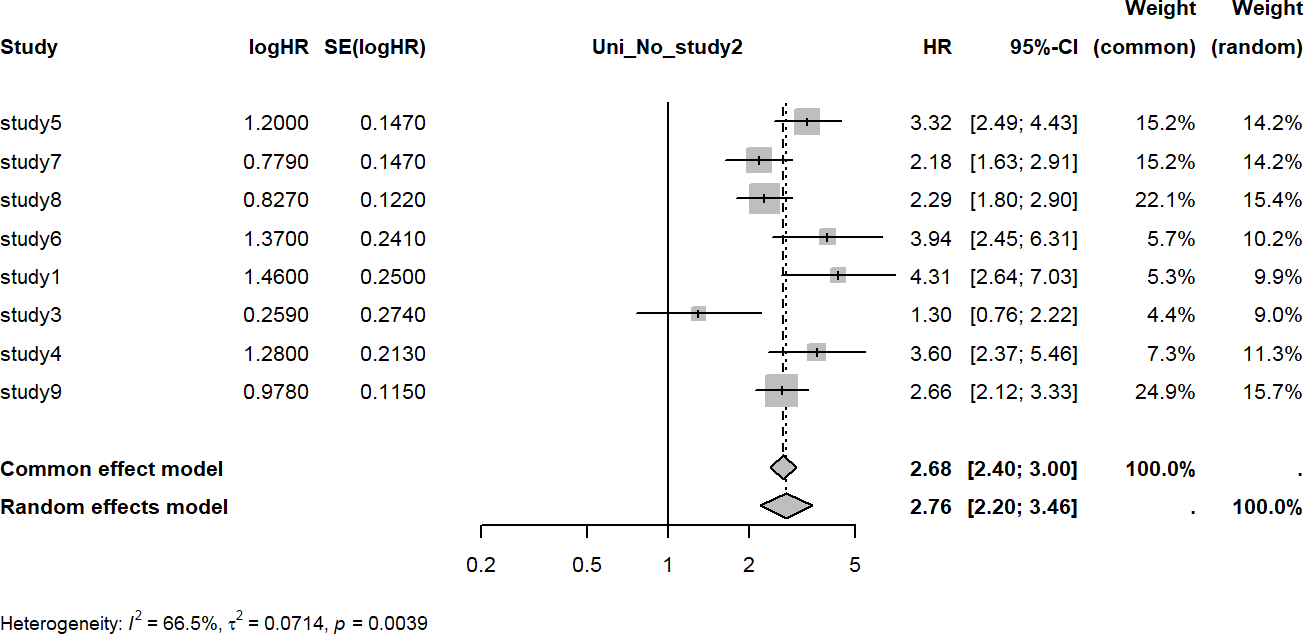
Source: Figure by author(s).

**Figure 92**. Study left out.



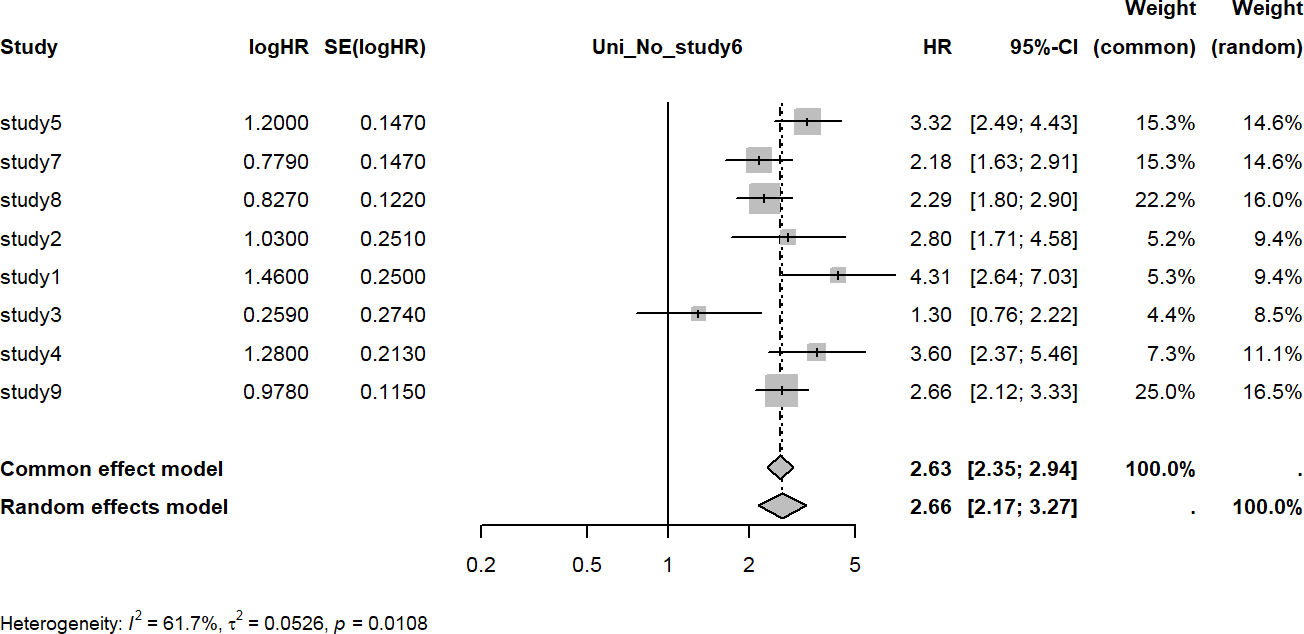
Source: Figure by author(s).

**Figure 93**. Study left out.



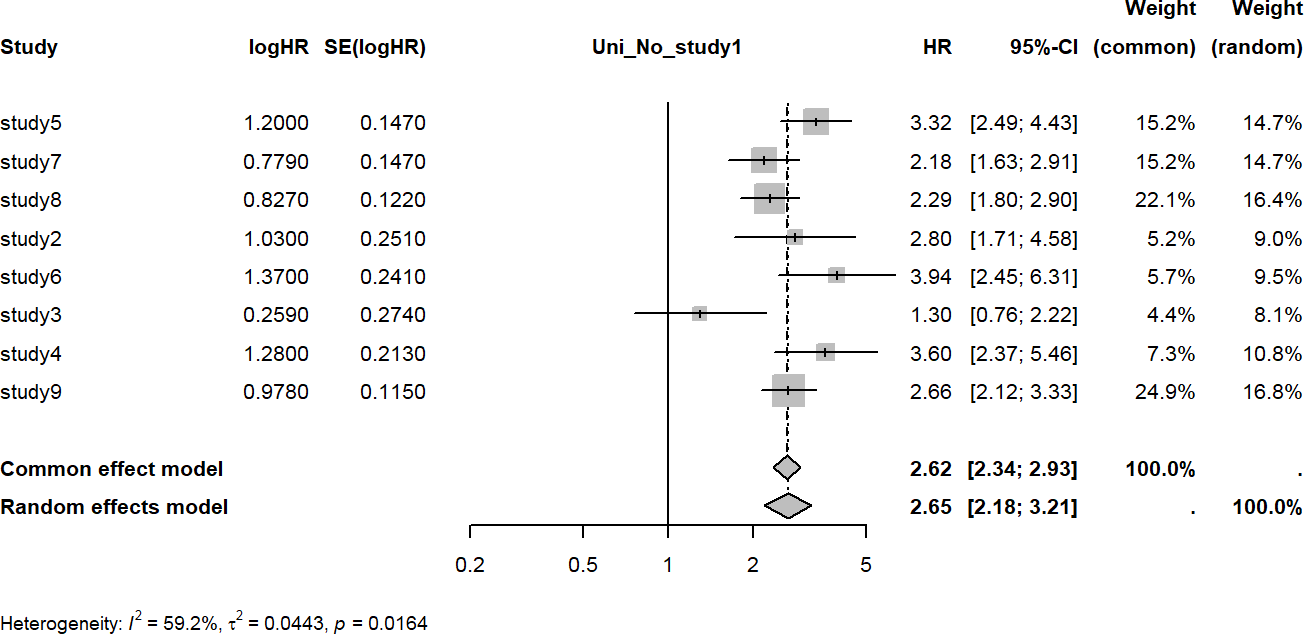
Source: Figure by author(s).

**Figure 94**. Study left out.



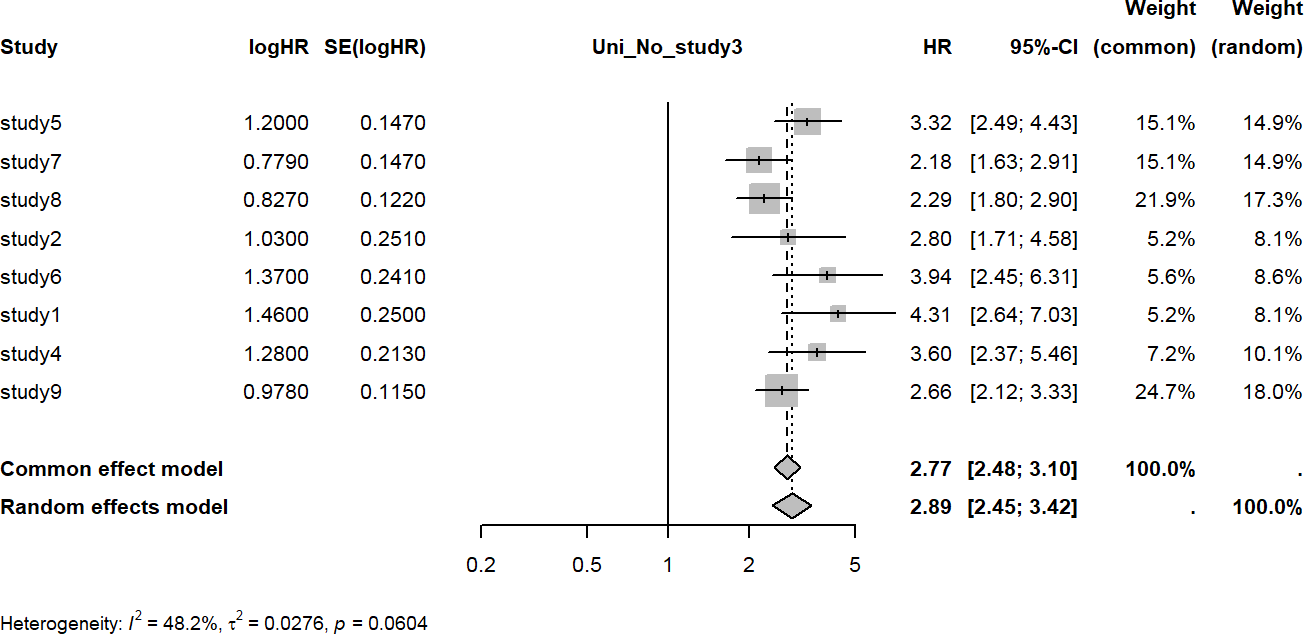
Source: Figure by author(s).

**Figure 95**. Study left out.



Source: Figure by author(s).

**Figure 96**. Study left out.



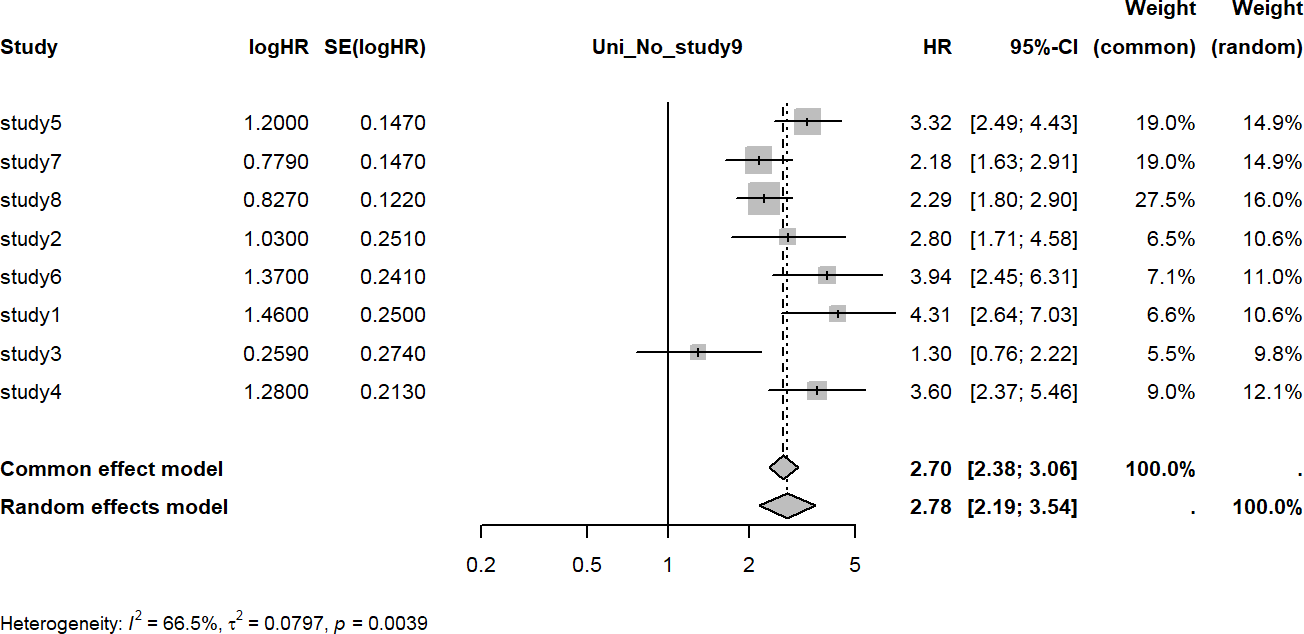
Source: Figure by author(s).

**Figure 97**. Study left out.



Source: Figure by author(s).

**Figure 98**. Study left out.



Source: Figure by author(s).

**Figure 99**. Study left out.

Now, we combine the results from the above leave one study out meta-analysis above.

v.mat<-as.data.frame(cbind(cname,hr.fixed,ll.fixed,ul.fixed,

hr.random,ll.random,ul.random)) v.mat$cname<-paste("No",v.mat$cname, sep = " ") v.mat[,c("hr.fixed","ll.fixed","ul.fixed",

"hr.random","ll.random","ul.random")]<-

lapply(v.mat[,c("hr.fixed","ll.fixed","ul.fixed",

"hr.random","ll.random","ul.random")],as.numeric)

v.mat[,c("hr.fixed","ll.fixed","ul.fixed",

"hr.random","ll.random","ul.random")]<-

round(v.mat[,c("hr.fixed","ll.fixed","ul.fixed",

"hr.random","ll.random","ul.random")],2)

*# Reset margins to original settings*

par(mar = old\_mar)

Then, we create a summary plot for leave one study out validation for random effect meta-analysis. The vertical dash line shows the original meta-analysis results of all studies)

ggplot(data = v.mat, aes(x = hr.random,y = cname))+

geom\_point() +

scale\_y\_discrete()+

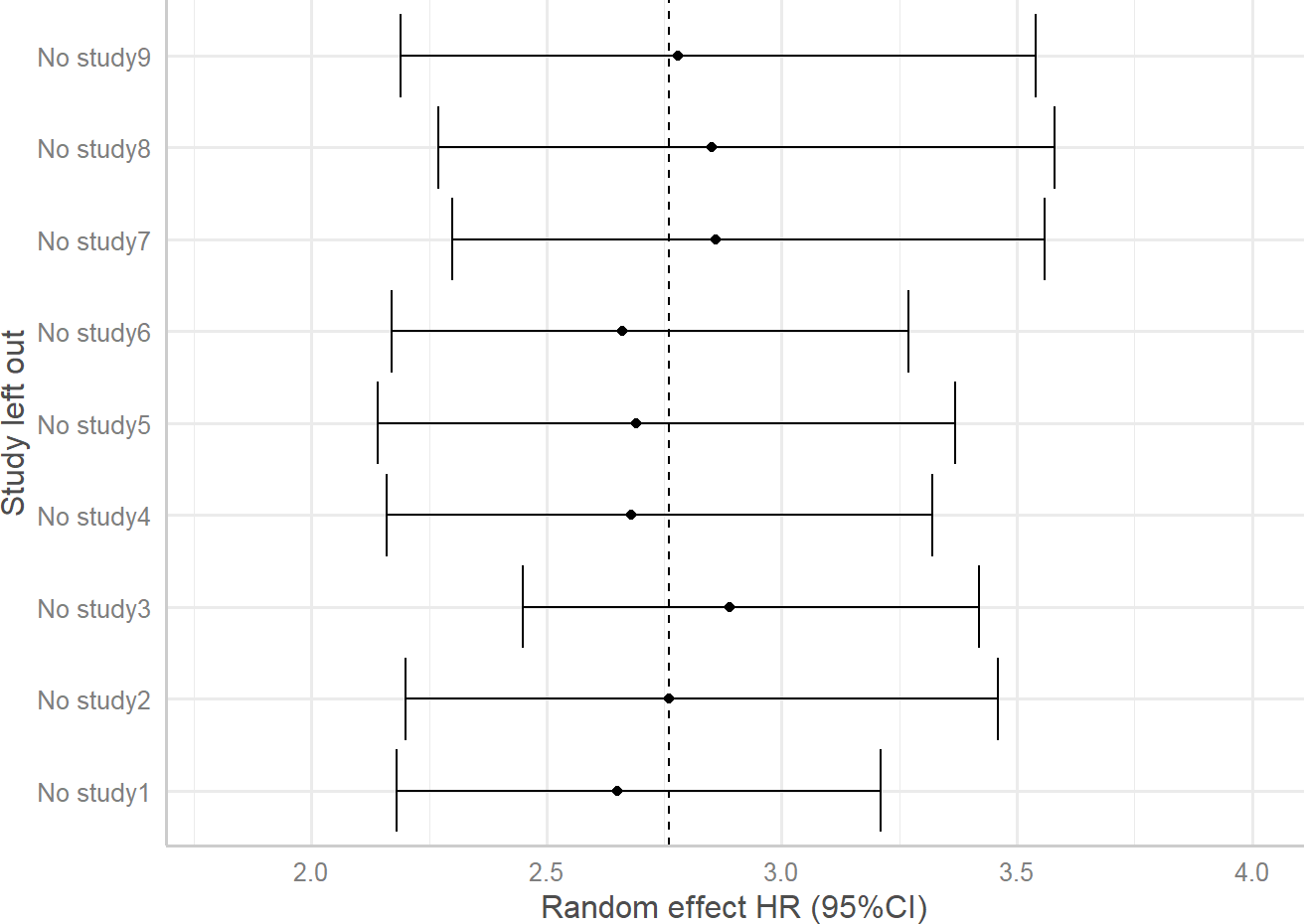
geom\_errorbar(aes(xmin = ll.random,xmax = ul.random))+

geom\_vline(xintercept = exp(mcfit$TE.random),

linetype = "dashed")+ xlab("Random effect HR (95%CI)")+

ylab("Study left out")+

xlim(1.8,4)



Source: Figure by author(s).

**Figure 100**. Sensitivity analysis summary: univariable.

This chapter provides example R codes for analysis and meta-analysis of large consortium data with example R functions and loops for more efficient analysis and result summary. One may apply and elaborate the example R codes in this chapter for their large, complex data.

## Useful Resources

The paper “Diabetes is associated with increased liver cancer incidence and mortality in adults: A report from Asia Cohort Consortium” published on [International Journal of Cancer by Nhan Thi Ho et al](https://doi.org/10.1002/ijc.34965) [61].

## Chapter Summary

* *tableby()* function of the “*arsenal”* package is handy for data summary
* *“survival”* and “*survminer”* packages are essential for survival analysis
* Newly created *subhe()* function may be used for subgroup analysis, heterogeneity test and result summary as a table for consortium data from multiple studies
* Univariable and multivariable model may be done for each study and results

are summarized as a table

* Loops and functions may be used to make repeated analysis more efficient
* *“meta”* package is used for meta-analysis across studies
* Sensitivity analysis (leave one study out validation) to evaluate if any specific study strongly affects the overall results

# Analysis of Child Growth Data with R

Child growth data requires special analysis approaches. This chapter will guide through the analyses to generate similar results to those of the paper “Overweight & obesity epidemic, temporal trends and regional disparities in physical growth of Vietnamese children” by Nhan Thi Ho et al. The example anthropometric dataset used in this chapter was generated to mimic the real data used in the paper. The data include height, weight, BMI of children from 18 months to 18 years measured annually during 7 years. This is a longitudinal dataset with repeated measures of the same subjects. The data will be compared with World Health Organization (WHO) 2007 child growth reference data [68], [69]. WHO 2007 child growth reference data are provided in this chapter examples.

This chapter provides example R codes for child growth specific analysis including: summary of anthropometric measures and comparing between groups, plotting study anthropometric data with corresponding WHO reference data for visual comparison, calculating Z-scores for study anthropometric data based on WHO references using both available R packages and newly created R function dedicated for this work, performing nutritional status classification (e.g. obesity, wasting, stunting,…) based on Z-scores and estimating corresponding prevalence, and constructing growth curves based on the study data using the WHO Lambda-Mu-Sigma (LMS) method. Alongside, there are examples of longitudinal data analysis for temporal trend, as well as examples for writing R functions and loops for more efficient analysis and result display. This chapter focuses on how to do specific analyses and result display using R, not on result output interpretation.

## Data Summary

### Load Required Packages and Example Data

The main R packages introduced in this chapter includes:

* *“anthro”* [70] & “*zscorer*” [71]: Specialized packages that contain functions to easily calculate child growth Z-scores based on WHO standards.
* *“DescTools”* [72] & “*confintr”* [73]: Provide functions to calculate robust confidence intervals for medians (*MedianCI()*) and for the difference between medians (*ci\_median\_diff()*), which are better for skewed growth data than mean-based statistics.
* *“gamlss”* [74]: The Generalized Additive Models for Location, Scale, and Shape package. We use this for the advanced WHO LMS method to construct growth curves based on study data.
* *“geepack” [51]*: For GEE models to analyze longitudinal trends. "*sjPlot*" [54]: For plotting GEE model results.
* “*ggplot2”:*  for visualization [18].

Some other supporting R packages (e.g. *"knitr" [5],"rmarkdown" [6],"kableExtra" [12], "tidyverse" [11], "rio" [14], "arsenal" [19], "gridExtra" [62], "hrbrthemes" [75], "MASS" [57], "viridis" [76], "emmeans" [77], "ggpubr" [20], "ggpmisc" [67], [78], "patchwork" [67], "stringr"* [79]) will also be used.

*# load multiple packages*

Packages < - c("knitr", "rmarkdown",

"kableExtra", "tidyverse",

"rio", "arsenal", "gridExtra",

"anthro", "zscorer", "hrbrthemes",

"MASS", "gamlss", "viridis", "confintr",

"DescTools", "geepack", "emmeans", "sjPlot",

"ggpubr","ggpmisc","patchwork", "stringr")

lapply(Packages, library, character.only = TRUE)

*# Data path*

url\_base < - "https://raw.githubusercontent.com/nhanhocu/"

data\_path < - "biodata-r/main/growdat.rda"

data\_url < - paste0(url\_base, data\_path)

*# load datasets*

con < - url(data\_url) load(con)

close(con)

*# quick look at example data*

str(dat)

*# quick look at WHO 2007 reference data for children 0-5 years old*

names(who5)

str(who5[[2]])

*# quick look at WHO 2007 reference data for children 5-19 years old*

names(who619) str(who619[[2]])

### Data Summary by Study Sites

Similar to previous chapters, we use the R package “*arsenal”* for data summary by study sites [19].

my\_controls < - tableby.control(

test = T,

total = T,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = c("meansd", "medianq1q3","meanCI","range","Nmiss2"),

cat.stats = c("countpct", "Nmiss2"),

stats.labels = list(

meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)",

range = "Min - Max",

Nmiss2 = "Missing",

meanCI = "Mean (95%CI)"

)

)

svar < -c("age","age.cat","sex","year")

mylabels < -as.list(svar)

names(mylabels) < -svar

tabs < - tableby(as.formula(paste("site",paste(svar[!svar %in% "site"],

collapse="+"), sep="~")),

data = dat,

control = my\_controls)

kable(summary(tabs,labelTranslations = mylabels, text=TRUE),

caption="Data summary by site")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 19**. Data summary by study site.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Site1 (N = 30355)** | **Site2 (N = 58191)** | **Site3 (N = 8791)** | **Total (N = 97337)** | ***p* Value** |
| age |  |  |  |  | < 0.001 |
| - Mean (SD) | 9.372 (3.822) | 8.696 (3.917) | 9.404 (3.786) | 8.971 (3.891) |  |
| - Median (Q1, Q3) | 9.200 (6.400, 12.200) | 8.400 (5.500, 11.600) | 9.400 (6.400, 12.200) | 8.800 (5.900, 11.900) |  |
| - Mean (95%CI) | 9.372 (9.329, 9.415) | 8.696 (8.664, 8.728) | 9.404 (9.324, 9.483) | 8.971 (8.946, 8.995) |  |
| - Min–Max | 1.500–18.300 | 1.500–18.200 | 1.500–17.900 | 1.500–18.300 |  |
| - Missing | 0 | 0 | 0 | 0 |  |
| age.cat |  |  |  |  | < 0.001 |
| - age 0–5 years | 4521 (14.9%) | 12,323 (21.2%) | 1322 (15.0%) | 18,166 (18.7%) |  |
| - age 6–11 years | 15,445 (50.9%) | 29,125 (50.1%) | 4380 (49.8%) | 48,950 (50.3%) |  |
| - age ≥12 years | 10,389 (34.2%) | 16,743 (28.8%) | 3089 (35.1%) | 30,221 (31.0%) |  |
| - Missing | 0 | 0 | 0 | 0 |  |

**Table 19.** *Cont.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Site1 (N = 30355)** | **Site2 (N = 58191)** | **Site3 (N = 8791)** | **Total (N = 97337)** | ***p* Value** |
| sex |  |  |  |  | < 0.001 |
| - Female | 15,142 (49.9%) | 27,703 (47.6%) | 4010 (45.6%) | 46,855 (48.1%) |  |
| - Male | 15,213 (50.1%) | 30,488 (52.4%) | 4781 (54.4%) | 50,482 (51.9%) |  |
| - Missing | 0 | 0 | 0 | 0 |  |
| year |  |  |  |  | < 0.001 |
| - 0 | 2550 (8.4%) | 506 (0.9%) | 405 (4.6%) | 3461 (3.6%) |  |
| - 1 | 4204 (13.8%) | 4200 (7.2%) | 738 (8.4%) | 9142 (9.4%) |  |
| - 2 | 3952 (13.0%) | 10,652 (18.3%) | 1331 (15.1%) | 15,935 (16.4%) |  |
| - 4 | 8570 (28.2%) | 14,244 (24.5%) | 2760 (31.4%) | 25,574 (26.3%) |  |
| - 5 | 5612 (18.5%) | 14,781 (25.4%) | 1838 (20.9%) | 22,231 (22.8%) |  |
| - 6 | 5467 (18.0%) | 13,808 (23.7%) | 1719 (19.6%) | 20,994 (21.6%) |  |
| - Missing | 0 | 0 | 0 | 0 |  |

Source: Table by author(s).

## Analysis Approaches

### Compare Median of Anthropometric Measure for Age by Gender

#### Define new functions for “*ggplot2*”

Below is a new function *iqr()* to calculate median and range at given quantiles. This function supports easy plotting the quantiles of choice with “*ggplot2”*. The default is to plot median and interquartile (IQR) range (*lower = 0.25, upper = 0.75*).

iqr = function(z, lower = 0.25, upper = 0.75) {

data.frame(

y = median(z),

ymin = quantile(z, lower),

ymax = quantile(z, upper)

)

}

Below is a new function *medci()* to calculate median and 95% confidence interval (95%CI) and support easy plotting the median and 95%CI with “*ggplot2”*.

medci < -function(z,ci=0.95){

data.frame(

y = median(z),

ymin = MedianCI(z, conf.level = ci, na.rm=TRUE,

method="exact")[2],

ymax = MedianCI(z, conf.level = ci, na.rm=TRUE,

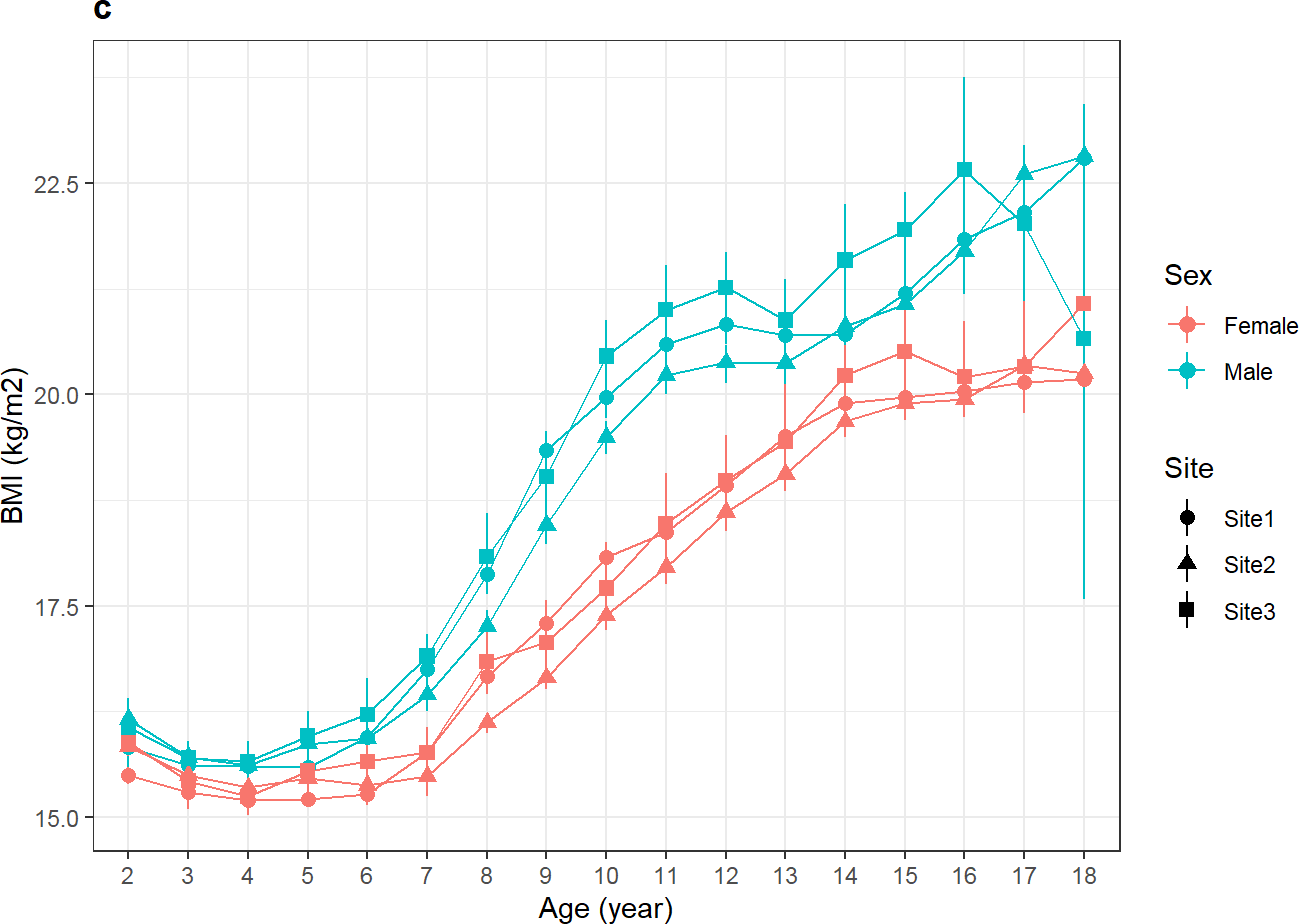
method="exact")[3]

)

}

#### Plot the Median, 95%CI Applying the New Function between Study Sites

The above functions are designed to work with “*ggplot2”'s stat\_summary()* layer. We now plot the median, 95%CI of BMI by age (in year), sex and study site with “*ggplot2”* and use the function *medci*() created above: *aes(..., group = interaction(sex, site))* tells *ggplot()* to create a separate line and summary for each unique combination of sex and site; *stat\_summary(fun.data = medci)* calls our *medci()* function for each group at each age point. It uses the returned *y, ymin*, and *ymax* values to plot the median (as a point) and its 95% CI (as a vertical error bar).



ggplot(dat, aes(x=age.y,

y=bmi,

color=sex,

shape=site,

group=interaction(sex,site))) +

stat\_summary(

fun = median,

geom='line') +

stat\_summary(

fun=median,

geom='point') +

stat\_summary(

fun.data = medci)+

theme\_bw()+

ggtitle("c")+

theme(plot.title = element\_text(face = "bold"))+

labs(x="Age (year)",y="BMI (kg/m2)",

color="Sex", shape="Site")

Source: Figure by author(s).

**Figure 101**. BMI median, 95%CI between study sites.

One may plot median and quantile range using the *iqr()* function above in place of the *medci()* function.

#### Calculate Median, 95%CI of Height

Median, 95%CI of anthropometric measures may be calculated using the function *MedianCI()* of the “*DescTools”* package [72]. The example below calculates median, 95%CI of height for males each age by writing a loop (repeating the calculation for each age) and combining results into a table.

datme < -dat[dat$sex %in% "Male",]

hk < -NULL

for (k in 2:18){

h < -MedianCI(datme$height[datme$age.y %in% k], conf.level = 0.95, na.rm=TRUE, method="exact")

hk < -rbind(hk,h)

}

age < -as.character(2:18)

hk < -round(hk,2)

hk < -cbind(age, hk)

rownames(hk) < -NULL

hb < - as.data.frame(cbind(age=age, median.ci=paste(hk[,2]," (",hk[,3],", ",hk[,4],")", sep="")))

kable(hb, caption="Height median, 95\\%CI of males")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 20**. Height median, 95%CI of males.

|  |  |
| --- | --- |
| **Age** | **Median.ci** |
| 2 | 87 (87, 87.5) |
| 3 | 95 (95, 95) |
| 4 | 102.5 (102, 102.6) |
| 5 | 110 (109.5, 110) |
| 6 | 117 (117, 117) |
| 7 | 123.5 (123, 123.6) |
| 8 | 129.3 (129, 129.5) |
| 9 | 134.8 (134.5, 135) |
| 10 | 140 (140, 140.4) |
| 11 | 146 (145.6, 146) |
| 12 | 153 (152.6, 153) |
| 13 | 161 (160.8, 161.4) |
| 14 | 166.35 (166, 166.7) |
| 15 | 169.5 (169.1, 170) |
| 16 | 171.2 (171, 171.8) |
| 17 | 172.7 (172, 173) |
| 18 | 172.75 (172, 174) |

Source: Table by author(s).

#### Calculate Median Difference (95%CI of Difference) Between Sites

The difference between medians and 95%CI of the difference may be calculated by two-sided 95% bootstrap confidence interval for the population value of median(x)-median(y) based on 999 bootstrap replications and the bias-corrected and accelerated (BCa) method implemented in the function *ci\_median\_diff()* of the “*confintr”* package [73]. The example below calculates median height difference between two sites (“Site2” vs. “Site3”) and 95%CI for median difference (“median.diff.ci”). We write a loop to repeat the calculation for each age and combine the results in a table.

datd < -subset(datme,site %in% c("Site2","Site3"))

a2 < -sort(unique(datd$age.y[datd$site %in% "Site2"]))

a3 < -sort(unique(datd$age.y[datd$site %in% "Site3"]))

ageu < -sort(a2[a2 %in% a3])

hk < -NULL

for (k in ageu){

h < -ci\_median\_diff(datd$height[datd$age.y %in% k & datd$site %in% "Site2"],

datd$height[datd$age.y %in% k & datd$site %in% "Site3"],

probs = c(0.025, 0.975),

type = "bootstrap",

boot\_type = "basic", *#CI by basic bootstrap method*

R = 999)

hd < -cbind(median.diff=h$estimate, l2.5=h$interval[1], l97.5=h$interval[2])

hk < -rbind(hk,hd)

}

hk < -round(hk,2)

rownames(hk) < -as.character(ageu)

hkb < - as.data.frame(cbind(age=as.character(ageu),

median.diff.ci=paste(hk[,1]," (",hk[,2],", ",hk[,3],")",

sep="")))

kable(hkb,

caption="Height median difference, 95\\%CI of males")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 21**. Height median difference, 95%CI of males.

**Age Median.diff.ci**

2 0.25 (−1.3, 1.35)

3 −0.5 (−1.7, −0.05)

4 −0.5 (−1, 0.5)

5 −1 (−1.5, 0)

|  |  |  |
| --- | --- | --- |
|  | **Table 21.** *Cont.* |  |
| **Age** |  | **Median.diff.ci** |
| 6 |  | −0.3 (−0.9, 0.2) |
| 7 |  | 0.2 (−1.1, 1.4) |
| 8 |  | 0.1 (−1.3, 0.2) |
| 9 |  | 0.5 (0, 1.6) |
| 10 |  | 0.25 (−0.9, 0.8) |
| 11 |  | 1 (0.4, 2.5) |
| 12 |  | 0.4 (−1.2, 1.4) |
| 13 |  | −0.4 (−1.85, 0.7) |
| 14 |  | −0.25 (−1.7, 0.8) |
| 15 |  | 0.5 (−0.7, 1.9) |
| 16 |  | 1.3 (0.9, 2.9) |
| 17 |  | 1.5 (0.65, 3.9) |
| 18 |  | −0.5 (−4.8, 4.7) |

Source: Table by author(s).

### Temporal Trend

Temporal trends over calendar years may be evaluated using Generalized Estimating Equations s (GEE) model with *geeglm()* of the “*geepack”* package, which evaluates average slope and allows accounting for repeated measures within subjects [51]. The correlation structure is selected based on the smallest Quasi Information Criterion (QIC). First, we fit the GEE model to evaluate how child height by age is changing over calendar year (“yearvisit”). The model includes “height” as dependent variable, “yearvisit”, “age.y”, and the interaction term between “yearvisit” and “age.y” (*age.y\*yearvisit*) as independent variables. With the interaction term, we want to examine if the temporal trend (“yearvisit” slope) depends on the child's age group (“age.y”). The example GEE model below uses “*independent*” correlation structure. Then we use the function *QIC()* to evaluate the QIC of the model.

datme$age.y < -droplevels(datme$age.y)

ghm < -geeglm(height ~ yearvisit +age.y+age.y\*yearvisit,

corstr = "independence",

id=cid, data=datme)

summary(ghm)

##

## Call:

## geeglm(formula = height ~ yearvisit + age.y + age.y \* yearvisit,

## data = datme, id = cid, corstr = "independence")

##

## Coefficients:

|  |  |  |  |
| --- | --- | --- | --- |
| ## | Estimate | Std.err | Wald |
| ## (Intercept) | 88.6409 | 0.2473 | 128481.84 |
| ## yearvisit | -0.4663 | 0.0636 | 53.67 |
| ## age.y3 | 7.3611 | 0.3045 | 584.57 |
| ## age.y4 | 14.8465 | 0.3053 | 2365.22 |
| ## age.y5 | 21.9698 | 0.3279 | 4489.25 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | age.y6 | 29.2578 | | 0.3129 | 8745.57 | |
| ## | age.y7 | 35.5397 | | 0.3199 | 12341.02 | |
| ## | age.y8 | 41.6244 | | 0.3207 | 16845.89 | |
| ## | age.y9 | 46.5552 | | 0.3481 | 17889.25 | |
| ## | age.y10 | 51.6295 | | 0.3512 | 21615.61 | |
| ## | age.y11 | 57.3249 | | 0.3785 | 22942.34 | |
| ## | age.y12 | 63.3965 | | 0.4291 | 21828.09 | |
| ## | age.y13 | 71.2617 | | 0.4646 | 23527.04 | |
| ## | age.y14 | 76.7558 | | 0.4397 | 30469.45 | |
| ## | age.y15 | 80.6519 | | 0.5054 | 25463.96 | |
| ## | age.y16 | 82.6985 | | 0.5720 | 20901.04 | |
| ## | age.y17 | 83.2308 | | 0.7019 | 14059.48 | |
| ## | age.y18 | 83.0227 | | 1.4546 | 3257.60 | |
| ## | yearvisit:age.y3 | 0.2112 | | 0.0782 | 7.30 | |
| ## | yearvisit:age.y4 | 0.2111 | | 0.0764 | 7.63 | |
| ## | yearvisit:age.y5 | 0.2434 | | 0.0816 | 8.90 | |
| ## | yearvisit:age.y6 | 0.2253 | | 0.0794 | 8.05 | |
| ## | yearvisit:age.y7 | 0.2417 | | 0.0801 | 9.11 | |
| ## | yearvisit:age.y8 | 0.1850 | | 0.0806 | 5.27 | |
| ## | yearvisit:age.y9 | 0.3752 | | 0.0840 | 19.94 | |
| ## | yearvisit:age.y10 | 0.4814 | | 0.0851 | 32.00 | |
| ## | yearvisit:age.y11 | 0.5269 | | 0.0912 | 33.37 | |
| ## | yearvisit:age.y12 | 0.7030 | | 0.1002 | 49.26 | |
| ## | yearvisit:age.y13 | 0.6274 | | 0.1087 | 33.30 | |
| ## | yearvisit:age.y14 | 0.6022 | | 0.1024 | 34.59 | |
| ## | yearvisit:age.y15 | 0.5010 | | 0.1143 | 19.21 | |
| ## | yearvisit:age.y16 | 0.4676 | | 0.1242 | 14.18 | |
| ## | yearvisit:age.y17 | 0.6111 | | 0.1442 | 17.96 | |
| ## | yearvisit:age.y18 | 0.7037 | | 0.2848 | 6.11 | |
| ## |  |  | Pr(>|W|) | | |  |
| ## | (Intercept) | < | 0.0000000000000002 | | | \*\*\* |
| ## | yearvisit |  | 0.00000000000024 | | | \*\*\* |
| ## | age.y3 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y4 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y5 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y6 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y7 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y8 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y9 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y10 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y11 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y12 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y13 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y14 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y15 | < | 0.0000000000000002 | | | \*\*\* |
| ## | age.y16 | < | 0.0000000000000002 | | | \*\*\* |

## age.y17 < 0.0000000000000002 \*\*\*

## age.y18 < 0.0000000000000002 \*\*\*

## yearvisit:age.y3 0.00690 \*\*

## yearvisit:age.y4 0.00574 \*\*

## yearvisit:age.y5 0.00285 \*\*

## yearvisit:age.y6 0.00456 \*\*

## yearvisit:age.y7 0.00255 \*\*

## yearvisit:age.y8 0.02168 \*

## yearvisit:age.y9 0.00000799747269 \*\*\*

## yearvisit:age.y10 0.00000001538845 \*\*\*

## yearvisit:age.y11 0.00000000762262 \*\*\*

## yearvisit:age.y12 0.00000000000224 \*\*\*

## yearvisit:age.y13 0.00000000789146 \*\*\*

## yearvisit:age.y14 0.00000000406950 \*\*\*

## yearvisit:age.y15 0.00001168611859 \*\*\*

## yearvisit:age.y16 0.00017 \*\*\*

## yearvisit:age.y17 0.00002258536144 \*\*\*

## yearvisit:age.y18 0.01348 \*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Correlation structure = independence

## Estimated Scale Parameters:

##

## Estimate Std.err

## (Intercept) 35.9 0.574

## Number of clusters: 16863 Maximum cluster size: 10

*# Examine QIC*

QIC(ghm)

## QIC QICu Quasi Lik CIC params QICC ## 1804358.3 1804356.1 -902144.0 35.1 34.0 1804358.4

One may refit the GEE model with another correlation structure and compare the QIC between GEE models to select the correlation structure with smallest QIC. In this case, “independence” structure has the smallest QIC and thus it is chosen.

Next, we get slopes and 95%CI for trend over years by age using *lstrends()* and display results in a table.

myModelSlopes < - lstrends(ghm, "age.y", var="yearvisit") kable(myModelSlopes, caption="Slopes raw table")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 22**. Slopes raw table.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Age.y** | **Yearvisit.Trend** | **SE** | **df** | **Lower.CL** | **Upper.CL** |
| 2 | −0.466 | 0.064 | 50,247 | −0.591 | −0.342 |
| 3 | −0.255 | 0.060 | 50,247 | −0.372 | −0.138 |
| 4 | −0.255 | 0.047 | 50,247 | −0.347 | −0.163 |
| 5 | −0.223 | 0.050 | 50,247 | −0.321 | −0.125 |
| 6 | −0.241 | 0.045 | 50,247 | −0.329 | −0.153 |
| 7 | −0.225 | 0.046 | 50,247 | −0.316 | −0.134 |
| 8 | −0.281 | 0.046 | 50,247 | −0.372 | −0.191 |
| 9 | −0.091 | 0.055 | 50,247 | −0.198 | 0.016 |
| 10 | 0.015 | 0.056 | 50,247 | −0.096 | 0.126 |
| 11 | 0.061 | 0.065 | 50,247 | −0.067 | 0.189 |
| 12 | 0.237 | 0.077 | 50,247 | 0.085 | 0.388 |
| 13 | 0.161 | 0.088 | 50,247 | −0.012 | 0.334 |
| 14 | 0.136 | 0.080 | 50,247 | −0.021 | 0.293 |
| 15 | 0.035 | 0.095 | 50,247 | −0.151 | 0.221 |
| 16 | 0.001 | 0.107 | 50,247 | −0.208 | 0.210 |
| 17 | 0.145 | 0.129 | 50,247 | −0.109 | 0.398 |
| 18 | 0.237 | 0.278 | 50,247 | −0.307 | 0.781 |

Source: Table by author(s).

Then we can create a clean table containing only the useful results that we want to display (slopes and 95%CI).

sld < -as.data.frame(myModelSlopes)

sld[,c(2,3,5,6)] < -round(sld[,c(2,3,5,6)],2)

slc.hm < -as.data.frame(cbind(Age = as.character(sld[,"age.y"]),

Slope.95CI = paste(sld[,"yearvisit.trend"], " (",

sld[,"lower.CL"], "; ",

sld[,"upper.CL"], ")",

sep="")))

kable(slc.hm,

caption="GEE slopes and 95\\%CI")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 23**. GEE slopes and 95%CI.

|  |  |
| --- | --- |
| **Age** | **Slope.95CI** |
| 2 | −0.47 (−0.59; −0.34) |
| 3 | −0.26 (−0.37; −0.14) |
| 4 | −0.26 (−0.35; −0.16) |
| 5 | −0.22 (−0.32; −0.13) |
| 6 | −0.24 (−0.33; −0.15) |
| 7 | −0.22 (−0.32; −0.13) |
| 8 | −0.28 (−0.37; −0.19) |
| 9 | −0.09 (−0.2; 0.02) |
| 10 | 0.02 (−0.1; 0.13) |
| 11 | 0.06 (−0.07; 0.19) |
| 12 | 0.24 (0.09; 0.39) |
| 13 | 0.16 (−0.01; 0.33) |
| 14 | 0.14 (−0.02; 0.29) |
| 15 | 0.03 (−0.15; 0.22) |
| 16 | 0 (−0.21; 0.21) |
| 17 | 0.14 (−0.11; 0.4) |
| 18 | 0.24 (−0.31; 0.78) |

Source: Table by author(s).

Next, we can turn the table above to a plot object (using *geom = "table"*) to combine with another plot object later in “*ggplot2”*:

gg.slc.hm < - ggplot() + theme\_void() +

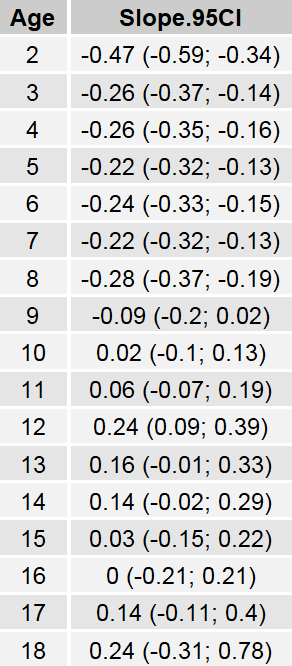
annotate(geom = "table",

x = 1,

y = 1,

label = list(slc.hm))

gg.slc.hm



Source: Figure by author(s).

**Figure 102**. GEE slopes and 95%CI.

Next, we visualize the trend slopes and 95%CI with the function *plot\_model()* of the *sjPlot*  package [54].

gee.hm < -plot\_model(ghm, type="int",

show.data=T,

show.values = TRUE,

show.p = TRUE,

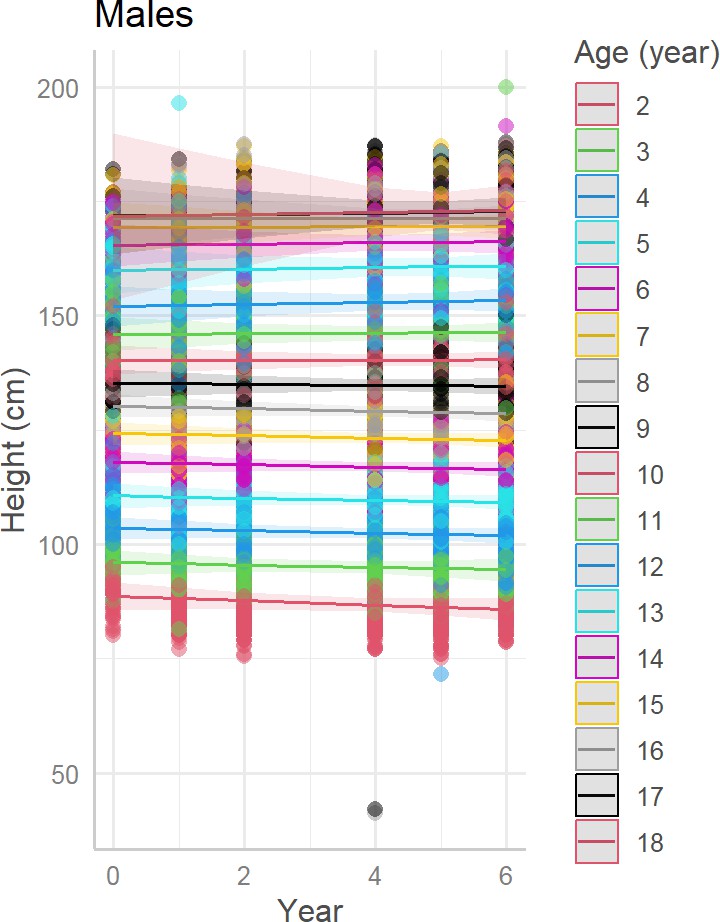
axis.title =c("Year","Height (cm)"),

title = "Males",

legend.title = "Age (year)",

colors = as.character(seq(2,18,by=1)))

gee.hm



Source: Figure by author(s).

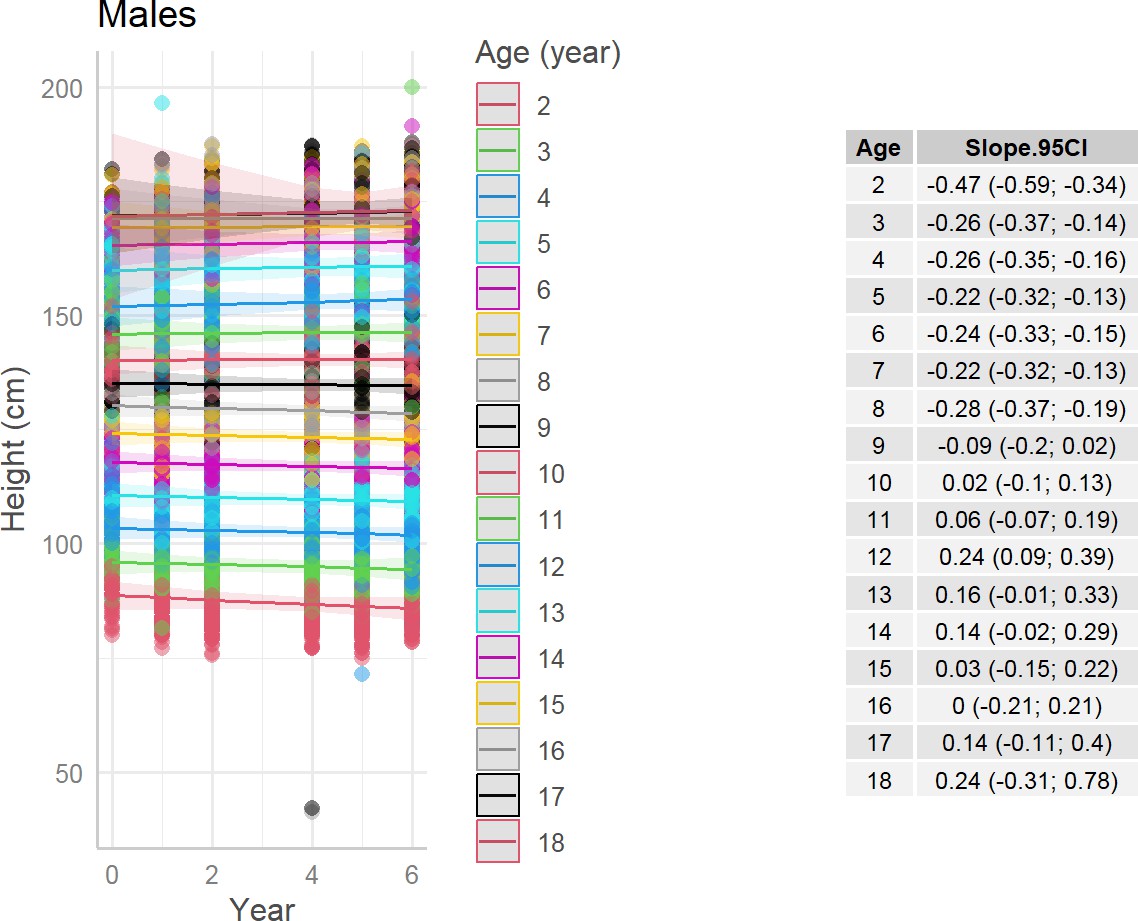
**Figure 103**. Trend slopes, 95%CI.

Then, we can combine the table plot object above with the slope plot object into one plot using *ggarrange()*:

gee\_hwb < -ggarrange(gee.hm, gg.slc.hm,

ncol=2, nrow=1, legend="right")

gee\_hwb



Source: Figure by author(s).

**Figure 104**. Trend slopes, 95%CI combined.

Note that for longitudinal growth trajectory, linear mixed effect model may be used.

### Comparison with WHO 2007 Growth References

#### Plot Study Data Against WHO References

First, we set color code, categories for height, weight, BMI and nutritional status classification:

colors2 < -c("Study median" = "red", "WHO median"="green", "+4SD"="blue",

"+3SD"="blue",

"+2SD"="blue",

"+1SD"="blue",

"-1SD"="purple",

"-2SD"="purple",

"-3SD"="purple",

"-4SD"="purple")

colors2i < -c("Observed" = "red",

"WHO median"="green", "+4SD"="blue",

"+3SD"="blue",

"+2SD"="blue",

"+1SD"="blue",

"-1SD"="purple",

"-2SD"="purple",

"-3SD"="purple",

"-4SD"="purple")

hl < -c("height >+4SD",

"+3SD < height~<~=+4SD",

"+2SD < height~<~=+3SD",

"+1SD < height~<~=+2SD",

"-1SD < = height~<~=+1SD",

"-2SD < = height~<~-1SD",

"-3SD < = height~<~-2SD",

"-4SD < = height~<~-3SD", "height~<~-4SD")

wl < -c("weight >+4SD",

"+3SD < weight~<~=+4SD",

"+2SD < weight~<~=+3SD",

"+1SD < weight~<~=+2SD",

"-1SD < = weight~<~=+1SD",

"-2SD < = weight~<~-1SD",

"-3SD < = weight~<~-2SD",

"-4SD < = weight~<~-3SD", "weight~<~-4SD")

bl < -c("BMI >+4SD",

"+3SD < BMI~<~=+4SD",

"+2SD < BMI~<~=+3SD",

"+1SD < BMI~<~=+2SD",

"-1SD < = BMI~<~=+1SD",

"-2SD < = BMI~<~-1SD",

"-3SD < = BMI~<~-2SD",

"-4SD < = BMI~<~-3SD",

"BMI~<~-4SD")

whl < -c("WFH >+4SD",

"+3SD < WFH~<~=+4SD",

"+2SD < WFH~<~=+3SD",

"+1SD < WFH~<~=+2SD",

"-1SD < = WFH~<~=+1SD",

"-2SD < = WFH~<~-1SD",

"-3SD < = WFH~<~-2SD",

"-4SD < = WFH~<~-3SD", "WFH~<~-4SD")

stl < -c("Extreme tallness","Normal HFA","Stunting")

w5l < -c("Obesity","Overweight","Normal WFH","Wasting")

t18l < -c("Obesity","Overweight","Normal BMI","Thinness")

WHO growth reference data are provided for ≤ 5 years old and ≥ 5 years old. Thus, in the example below, we get data for boys ≤ 5 years old from the total study dataset, get WHO reference height for age data for boys ≤ 5 and summarize by age in month:

dat05b < -dat[dat$age < =5 & dat$sex %in% "Male",]

hfa.boy.z.05 < -who5$`lhfa-boys-zscore-expanded-tables`

hfa.boy.z.05$age.m < -round(hfa.boy.z.05$day/30,0)

hfa.boy.z.05.m < -hfa.boy.z.05 %>%

group\_by(age.m)%>%

summarise(across(everything(), median))

Then, we plot study data median, 95%CI and actual data points against WHO median and standard deviation (SD) using our new function *medci()* and the color code created previously:

who.5hm < -ggplot() +

geom\_point(dat05b, mapping=aes(x=age.m, y=height), size=0.1)+

stat\_summary(dat05b, mapping=aes(x=age.m, y=height,

color="Study median"),

fun = median, geom='line') +

stat\_summary(dat05b, mapping=aes(x=age.m, y=height), fun=median,

geom='point', colour="red") +

stat\_summary(dat05b, mapping=aes(x=age.m, y=height),

fun.data=medci, geom='errorbar', width=0.2, colour="red") +

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd0, color="WHO median"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd1, color="+1SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd2, color="+2SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd3, color="+3SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd4, color="+4SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd1neg, color="-1SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd2neg, color="-2SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd3neg, color="-3SD"))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd4neg, color="-4SD"))+

theme\_bw()+

scale\_color\_manual(name = "", values = colors2,

breaks = names(colors2)[c(1:10)])+

theme\_bw()+

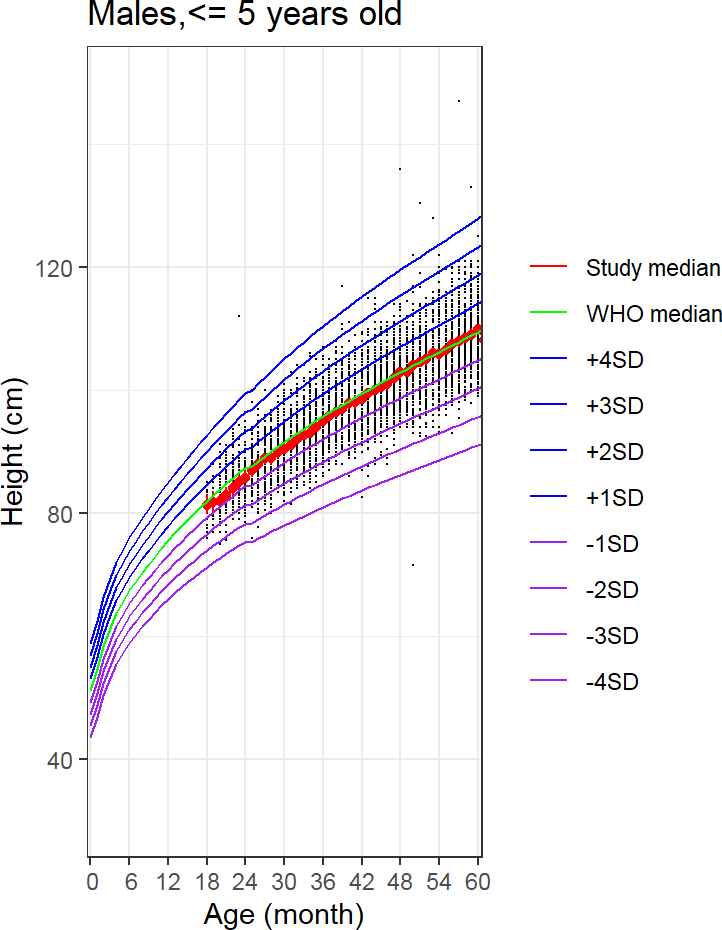
labs(x = "Age (month)", y="Height (cm)")+

coord\_cartesian(xlim = c(0, 60), ylim=c(30,150))+

scale\_x\_discrete(limits =seq(0, 60, by = 6))+

ggtitle("Males, < = 5 years old")

who.5hm



Source: Figure by author(s).

**Figure 105**. Study data height vs. WHO references.

#### Plot Individual Data Against WHO References

We can plot to examine the growth trajectory of an individual as compared to WHO reference. For example, we plot the data of a child with “cid” =“c23782”:

*# get an individual data*

datin < -dat05b[dat05b$cid %in% "c23782", ]

who.inh < - ggplot()+

geom\_point(datin, mapping=aes(x = age.m, y = height,

group = cid,

colour=”Observed”))+

geom\_line(datin, mapping=aes(x = age.m, y = height,

group = cid, colour=”Observed”), size=0.3)+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd0, color=”WHO median”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd1, color=”+1SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd2, color=”+2SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd3, color=”+3SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd4, color=”+4SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd1neg, color=”-1SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd2neg, color=”-2SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd3neg, color=”-3SD”))+

geom\_line(hfa.boy.z.05.m, mapping=aes(x=age.m, y=sd4neg, color=”-4SD”))+

theme\_bw()+

scale\_color\_manual(name = “”, values = colors2i,

breaks = names(colors2i)[c(1:10)])+

theme\_bw()+

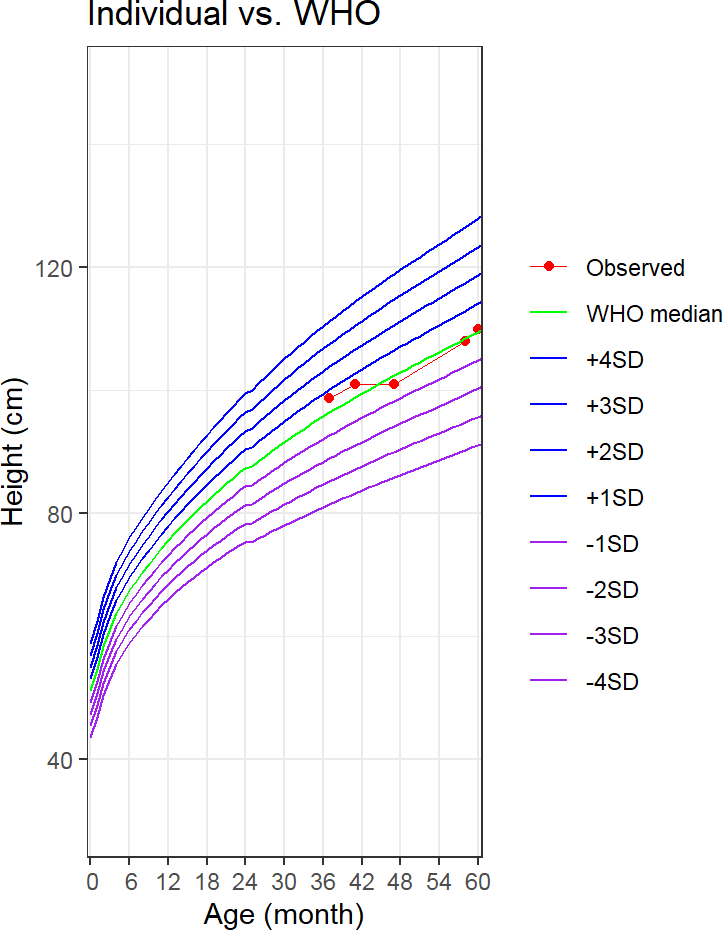
labs(x = “Age (month)”, y=”Height (cm)”)+

coord\_cartesian(xlim = c(0, 60), ylim=c(30,150))+

scale\_x\_discrete(limits =seq(0, 60, by = 6))+

ggtitle(“Individual vs. WHO”)

who.inh



Source: Figure by author(s).

**Figure 106**. Individual height vs. WHO references.

#### Nutritional Status Classification According to WHO References

One way to do this is to merge study data and WHO reference data by age in months, then create a new height classification variable (using *case\_when()* of “*tidyverse”*) [11]:

dat05b.h.o < -merge(dat05b, hfa.boy.z.05.m, by="age.m", all.x=TRUE)

dat05b.h.o < - dat05b.h.o %>%

mutate(

height.cat = case\_when(height>=sd1neg & height < =sd1 ~ "-1SD < = height <= +1SD",

height>sd1 & height < = sd2 ~ "+1SD < height <=+2SD",

height>sd2 & height < = sd3 ~ "+2SD < height <=+3SD",

height>sd3 & height < = sd4 ~ "+3SD < height <=+4SD",

height>sd4 ~"height >+4SD",

height>=sd2neg & height < sd1neg ~ "-2SD <= height < -1SD",

height>=sd3neg & height < sd2neg ~ "-3SD <= height < -2SD",

height>=sd4neg & height < sd3neg ~ "-4SD <= height < -3SD",

height < sd4neg ~"height < -4SD"))

dat05b.h.o$height.cat < -factor(dat05b.h.o$height.cat, levels=hl)

tbl < - table(dat05b.h.o$height.cat)

Then we can calculate the prevalence and 95%CI for prevalence of height classification using Glaz & Sison’s method for simultaneous confidence intervals for multinomial proportions using the function *MultinomCI()* of the “*DescTools”* package [72] and display the results in a table.

pci < -round(MultinomCI(tbl,

conf.level=0.95, method="sisonglaz"),5)

tblb < - as.data.frame(cbind(count=tbl, p.95ci=paste(pci[,1], " (",

pci[,2], ", ",

pci[,3], ")",

sep="")))

kable(tblb,

caption="Height classification prevalence and 95\\%CI")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 24**. Height classification prevalence and 95%CI.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| height > +4SD | 10 | 0.00104 (0, 0.01044) |
| +3SD < height ≤ +4SD | 25 | 0.00261 (0, 0.012) |
| +2SD < height ≤ +3SD | 208 | 0.02171 (0.01242, 0.03111) |
| +1SD < height ≤ +2SD | 1025 | 0.10699 (0.0977, 0.11639) |
| −1SD ≤ height ≤ +1SD | 6486 | 0.67704 (0.66775, 0.68643) |
| −2SD ≤ height < −1SD | 1559 | 0.16273 (0.15344, 0.17213) |
| −3SD ≤ height < −2SD | 245 | 0.02557 (0.01628, 0.03497) |
| −4SD ≤ height < −3SD | 20 | 0.00209 (0, 0.01148) |
| height < −4SD | 2 | 0.00021 (0, 0.0096) |

Source: Table by author(s).

We can classify height into nutritional status categories following WHO references and calculate the prevalence, and 95%CI of these categories (e.g. stunting).

dat05b.h.o < - dat05b.h.o %>%

mutate(

height.cats = case\_when(height> sd3 ~"Extreme tallness",

height>=sd2neg & height < = sd3 ~"Normal HFA",

height < sd2neg ~"Stunting"))

dat05b.h.o$height.cats < -factor(dat05b.h.o$height.cats, levels=stl)

tbl < - table(dat05b.h.o$height.cats)

pci < -round(MultinomCI(tbl,

conf.level=0.95, method="sisonglaz"),5)

tblb < - as.data.frame(cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep="")))

kable(tblb,

caption="Prevalence, 95\\%CI of stunting")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 25**. Prevalence, 95%CI of stunting.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Extreme tallness | 35 | 0.00365 (0.00042, 0.00702) |
| Normal HFA | 9278 | 0.96848 (0.96524, 0.97184) |
| Stunting | 267 | 0.02787 (0.02463, 0.03124) |

Source: Table by author(s).

The prevalence and 95%CI for BMI, weight, height classification as overweight, obesity, wasting or thinness and stunting may be done similarly.

### Calculate Z-Scores and Prevalence with “Anthro” Package

One may use the “*[anthro](https://cran.r-project.org/web/packages/anthro/anthro.pdf)*” package [70] to make it easier to standardize the measurements of height, weight, BMI into Z-scores (or Standard Deviation Scores) by comparing them to WHO references. Then one may classify the Z-scores as overweight, obesity, wasting or thinness following WHO recommendation and calculate their prevalence.

#### Calculate Z-Scores for a Dataset

We can calculate Z-scores for all subjects and many anthropometric indices in a dataset at a time. The *anthro\_zscores()* function requires age in days, ‘*lenhei*’ is the column for length/height, ‘*weight*’ for weight. Note that, the package “*anthro"* calculates Z-scores for age ≤ 5 years old and may produce NA for age > 5 years old. It is important that age should be in days and sex should be numeric with *1 = male* and *2 = female*.

*# turn sex to numeric*

dat < - dat %>%

mutate(sex\_numeric = case\_when(

sex == "Male" ~ 1,

sex == "Female" ~ 2,

TRUE ~ NA\_real\_ *# Handles any other cases*

)

)

*# get data for age~<~=5*

dat5 < -dat[dat$age.d < =365\*5,]

z5 < -with(dat5,

anthro\_zscores(

sex = sex\_numeric,

age = age.d,

weight = weight,

lenhei = height

)

)

str(z5)

## 'data.frame': 17763 obs. of 20 variables:

## $ clenhei : num 112 106 104 106 102 ...

## $ cbmi : num 17.5 16 18.7 18.2 17.5 ...

## $ cmeasure: chr NA NA NA NA ...

## $ csex : int 1 2 2 2 2 2 1 1 2 2 ...

## $ zlen : num 0.52 -0.45 -0.91 -0.59 -0.83 0.08 -0.63 -0.88 -0.1 1.39 ...

## $ flen : int 0 0 0 0 0 0 0 0 0 0 ...

## $ zwei : num 1.37 0.08 0.83 0.86 0.46 1.32 -0.78 -0.34 0.05 0.93 ...

## $ fwei : int 0 0 0 0 0 0 0 0 0 0 ...

## $ zwfl : num 1.46 0.53 2.15 1.88 1.49 1.91 -0.68 0.3 0.14 -0.03 ...

## $ fwfl : int 0 0 0 0 0 0 0 0 0 0 ...

## $ zbmi : num 1.58 0.49 1.97 1.74 1.39 1.86 -0.62 0.34 0.14 0.21 ...

## $ fbmi : int 0 0 0 0 0 0 0 0 0 0 ...

## $ zhc : num NA NA NA NA NA NA NA NA NA NA ...

## $ fhc : int NA NA NA NA NA NA NA NA NA NA ...

## $ zac : num NA NA NA NA NA NA NA NA NA NA ...

## $ fac : int NA NA NA NA NA NA NA NA NA NA ...

## $ zts : num NA NA NA NA NA NA NA NA NA NA ...

## $ fts : int NA NA NA NA NA NA NA NA NA NA ...

## $ zss : num NA NA NA NA NA NA NA NA NA NA ...

## $ fss : int NA NA NA NA NA NA NA NA NA NA ...

*# To add the z-scores to study dataset*

dz5 < -as\_tibble(dat5) %>%

drop\_na() %>% *# needs to have no NA values*

mutate(anthro\_zscores(sex = sex\_numeric,

age = age.d,

is\_age\_in\_month = FALSE,

weight = weight,

lenhei = height))

str(dz5)

## tibble [17,419 x 35] (S3: tbl\_df/tbl/data.frame)

## $ cid : Factor w/ 88678 levels "c1","c2","c3",..: 828 1230 1684 1726 1726 2132 2694 2845

3077 3182 ...

## $ sex : chr [1:17419] "Male" "Female" "Female" "Female" ...

## $ weight : num [1:17419] 22 18 20.2 20.5 18.2 21.9 16.5 16.8 17.7 20.6 ...

## $ height : num [1:17419] 112 106 104 106 102 ...

## $ bmi : num [1:17419] 17.5 16 18.7 18.2 17.5 ...

## $ site : Factor w/ 3 levels "Site1","Site2",..: 2 2 2 2 2 2 2 2 2 2 ...

## $ age : num [1:17419] 4.9 4.8 4.8 4.9 4.4 4.9 5 4.6 4.7 4.9 ...

## $ age.d : num [1:17419] 1807 1750 1758 1789 1621 ...

## $ age.m : num [1:17419] 59 57 58 59 53 58 60 55 56 58 ...

## $ age.y : Factor w/ 18 levels "2","3","4","5",..: 4 4 4 4 3 4 4 4 4 4 ...

## $ yearvisit : num [1:17419] 0 0 0 0 0 0 1 0 0 1 ...

## $ year : Factor w/ 6 levels "0","1","2","4",..: 1 1 1 1 1 1 2 1 1 2 ...

## $ age.cat : Factor w/ 3 levels "age 0-5 years",..: 1 1 1 1 1 1 1 1 1 1 ...

## $ bmi.cat : Factor w/ 8 levels "BMI<=10","10<BMI <=15",..: 3 3 3 3 3 3 2 3 3 3 ...

## $ sex\_numeric: num [1:17419] 1 2 2 2 2 2 1 1 2 2 ...

## $ clenhei : num [1:17419] 112 106 104 106 102 ...

## $ cbmi : num [1:17419] 17.5 16 18.7 18.2 17.5 ...

## $ cmeasure : chr [1:17419] NA NA NA NA ...

## $ csex : int [1:17419] 1 2 2 2 2 2 1 1 2 2 ...

## $ zlen : num [1:17419] 0.52 -0.45 -0.91 -0.59 -0.83 0.08 -0.63 -0.88 -0.1 1.39 ...

## $ flen : int [1:17419] 0 0 0 0 0 0 0 0 0 0 ...

## $ zwei : num [1:17419] 1.37 0.08 0.83 0.86 0.46 1.32 -0.78 -0.34 0.05 0.93 ...

## $ fwei : int [1:17419] 0 0 0 0 0 0 0 0 0 0 ...

## $ zwfl : num [1:17419] 1.46 0.53 2.15 1.88 1.49 1.91 -0.68 0.3 0.14 -0.03 ...

## $ fwfl : int [1:17419] 0 0 0 0 0 0 0 0 0 0 ...

## $ zbmi : num [1:17419] 1.58 0.49 1.97 1.74 1.39 1.86 -0.62 0.34 0.14 0.21 ...

## $ fbmi : int [1:17419] 0 0 0 0 0 0 0 0 0 0 ...

## $ zhc : num [1:17419] NA NA NA NA NA NA NA NA NA NA ...

## $ fhc : int [1:17419] NA NA NA NA NA NA NA NA NA NA ... ## $ zac : num [1:17419] NA NA NA NA NA NA NA NA NA NA ...

## $ fac : int [1:17419] NA NA NA NA NA NA NA NA NA NA ... ## $ zts : num [1:17419] NA NA NA NA NA NA NA NA NA NA ...

## $ fts : int [1:17419] NA NA NA NA NA NA NA NA NA NA ... ## $ zss : num [1:17419] NA NA NA NA NA NA NA NA NA NA ...

## $ fss : int [1:17419] NA NA NA NA NA NA NA NA NA NA ...

#### Calculate Z-Scores for Individuals

We can also use the *anthro\_zscores()* function to calculate Z-scores for an individual. For example, we calculate Z-scores for a male 700 days old with weight = 9 kg, height = 85 cm.

zi < -anthro\_zscores(sex = 1,

age = 700,

weight = 9,

lenhei = 85

)

zi

## clenhei cbmi cmeasure csex zlen flen zwei fwei zwfl fwfl

## 1 85 12.5 <NA> 1 -0.65 0 -2.5 0 -3.12 0

## zbmi fbmi zhc fhc zac fac zts fts zss fss

## 1 -3.26 0 NA NA NA NA NA NA NA NA

### Calculate Prevalence of Z-Score Categories with “anthro” Package

We can calculate the prevalence of Z-Score Categories using the *anthro\_prevalence()* function of the “*anthro”* package [70]:

pz5 < -with( dat5,

anthro\_prevalence(

sex = sex\_numeric,

age = age.d,

weight = weight,

lenhei = height

)

)

head(pz5[,1:5])

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | Group | | | | HAZ\_pop | HAZ\_unwpop | HA\_3\_r | HA\_3\_se |
| ## 1 | All | | | | 17642 | 17642 | 0.249 | 0.0376 |
| ## 2 | Age | group: | 00-05 | mo | 0 | 0 | NA | NA |
| ## 3 | Age | group: | 06-11 | mo | 0 | 0 | NA | NA |
| ## 4 | Age | group: | 12-23 | mo | 990 | 990 | 0.909 | 0.3017 |
| ## 5 | Age | group: | 24-35 | mo | 3959 | 3959 | 0.177 | 0.0668 |
| ## 6 | Age | group: | 36-47 | mo | 5781 | 5781 | 0.346 | 0.0772 |
|  |  |  |  |  |  |  |  |  |

### Calculate Prevalence of Nutritional Status Manually

If one does not like the default prevalence calculation of the “*anthro"* package, one may calculate prevalence manually from the Z-scores generated from the *anthro\_zscores()* function above. First, we get nutritional status by categorizing Z-scores (e.g. of weight-for-height “zwfl”) following WHO recommendation using the function *cut()*.

dz5 < -dz5 %>% mutate(wh.catw= cut(

zwfl,

breaks = c(-Inf,-2, 2, 3, Inf), *# Define break points*

labels = c("Wasting", "Normal WFH", "Overweight","Obesity"), *# labels*

right = TRUE

))%>%

mutate(height.cats= cut( zlen,

breaks = c(-Inf,-2, 3, Inf), *# Define break points*

labels = c("Stunting", "Normal HFA", "Extreme tallness"), *# labels*

right = TRUE))

Then we calculate the prevalence of wasting, obesity based on weight-for-height z-score categories and display the results in a table.

*#w5l < -c("Obesity","Overweight","Normal WFH","Wasting")*

dz5$wh.catw < -factor(dz5$wh.catw, levels=w5l)

tbl < - table(dz5$wh.catw)

pci < -round(MultinomCI(tbl,

conf.level=0.95,

method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep=""))

kable(tblb, caption="Prevalence of wasting, obesity based on z-scores")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 26**. Prevalence of wasting, obesity based on z-scores.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Obesity | 389 | 0.02237 (0.01852, 0.02632) |
| Overweight | 870 | 0.05003 (0.04618, 0.05398) |
| Normal WFH | 15,901 | 0.91443 (0.91058, 0.91838) |
| Wasting | 229 | 0.01317 (0.00932, 0.01712) |

Source: Table by author(s).

Similarly, we can calculate the prevalence of stunting based on height-for-age z-score categories and display the results in a table.

*#stl < -c("Extreme tallness","Normal HFA","Stunting")*

dz5$height.cats < -factor(dz5$height.cats, levels=stl)

tbl < - table(dz5$height.cats)

pci < -round(MultinomCI(tbl, conf.level=0.95, method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep=""))

kable(tblb, caption="Prevalence of stunting based on z-scores")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 27**. Prevalence of stunting based on z-scores.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Extreme tallness | 46 | 0.00264 (0, 0.00533) |
| Normal HFA | 16,778 | 0.9632 (0.96056, 0.96589) |
| Stunting | 595 | 0.03416 (0.03152, 0.03685) |

Source: Table by author(s).

### Calculate Z-Scores Manually

The packages “*anthro* [70], and “*zscorer”* [71] have built-in functions to calculate Z-scores (see examples above). However, they produce errors or NA for data of children > 5 years old and lack of flexibility. For those with some R programming skill, it may be handier to download the WHO reference data from WHO website and do some programming to calculate Z-scores, especially for children > 5 years old. Note that, there are two sets of WHO reference data for children 0 to 5 years old (age in days) and for children 5 to 19 years old (age in months). Now, look at the WHO reference dataset that the “*zscorer”* package uses. The dataset is named “wgsData”. It contains the column “indicator” (a vector of indices *“hfa”, “wfa”, “wfh”*), “*sex*” (as numeric with 1 for male and 2 for female), “*given*” (either age in days or height in cm), *“l”, “m”, “s”*.

data(wgsData)

str(wgsData)

## 'data.frame': 1746 obs. of 6 variables:

## $ indicator: Factor w/ 3 levels "hfa","wfa","wfh": 3 3 3 3 3 3 3 3 3 3 ...

## $ sex : num 1 1 1 1 1 1 1 1 1 1 ...

## $ given : num 45 45.1 45.2 45.3 45.4 45.5 45.6 45.7 45.8 45.9 ...

## $ l : num -0.352 -0.352 -0.352 -0.352 -0.352 ...

## $ m : num 2.44 2.46 2.47 2.49 2.51 ...

## $ s : num 0.0918 0.0918 0.0917 0.0916 0.0916 ...

#### Prepare WHO reference data

First, we create a combined WHO reference dataset for height, weight, BMI for age and weight for height by sex (numeric with *1 = male*, *2 = female*) similarly to the WHO reference dataset “wgsData” of the “*zscorer”* package. We combined separate WHO reference dataset for children 0–5 years old (“whoref5”) and for children 5–19 years old (“whoref19”). Note that age should be in days for children 0–5 years old and age should be in months for children 5–19 years old. Both datasets contain the crucial L, M, and S parameters for each age and sex.

*# Combine WHO reference data for children >5 years old*

zn < -names(who619)[grep("z",names(who619))]

whoref19 < -NULL

for (i in zn){

test < -as.data.frame(who619[[i]])

indicator < - str\_extract(i, paste0("ˆ[ˆ", "\_", "]+"))

test < -cbind(indicator=indicator, test)

test$sex < -ifelse(grepl("boys", i) ==TRUE,1,2)

colnames(test)[colnames(test) %in% "month"] < -"given"

whoref19 < -plyr::rbind.fill(whoref19,test)

}

whoref19$indicator[whoref19$indicator %in% "bmi"] < -"bfa"

whoref19 < -as.data.frame(whoref19)

str(whoref19)

## 'data.frame': 792 obs. of 17 variables:

## $ indicator: chr "bfa" "bfa" "bfa" "bfa" ...

## $ given : num 61 62 63 64 65 66 67 68 69 70 ...

## $ l : num -0.739 -0.762 -0.786 -0.809 -0.832 ...

## $ m : num 15.3 15.3 15.3 15.3 15.3 ...

## $ s : num 0.0839 0.0841 0.0844 0.0846 0.0849 ...

## $ sd4neg : num 11.2 11.2 11.2 11.2 11.2 ...

## $ sd3neg : num 12.1 12.1 12.1 12.1 12.1 ...

## $ sd2neg : num 13 13 13 13 13 ...

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | $ sd1neg : | | | num | 14.1 | 14.1 | 14.1 | | 14.1 | | 14.1 | ... |
| ## | $ sd0 : | | | num | 15.3 | 15.3 | 15.3 | | 15.3 | | 15.3 | ... |
| ## | $ sd1 : | | | num | 16.6 | 16.6 | 16.7 | | 16.7 | | 16.7 | ... |
| ## | $ sd2 : | | | num | 18.3 | 18.3 | 18.3 | | 18.3 | | 18.3 | ... |
| ## | $ sd3 : | | | num | 20.2 | 20.2 | 20.2 | | 20.3 | | 20.3 | ... |
| ## | $ sd4 : | | | num | 22.1 | 22.1 | 22.2 | | 22.2 | | 22.3 | ... |
| ## | $ | sex | : | num | 1 1 1 | 1 1 1 | | 1 1 1 | | 1 ... | |  |
| ## | $ | stdev | : | num | NA NA | NA NA | | NA NA | | NA NA | | NA NA ... |
| ## | $ | sd5neg | : | num | NA NA | NA NA | | NA NA | | NA NA | | NA NA ... |

*# study data > 5 years old*

dat18 < -dat[dat$age.m>=61,]

*# Combine WHO reference data for children <=5 years old*

zn5 < -names(who5)[grep("z",names(who5))]

zn5 < -zn5[grep("expanded", zn5)]

whoref5 < -NULL for (i in zn5){

test < -as.data.frame(who5[[i]])

indicator < - str\_extract(i, paste0("ˆ[ˆ", "-", "]+"))

test < -cbind(indicator=indicator, test)

test$sex < -ifelse(grepl("boys", i) ==TRUE,1,2)

colnames(test)[colnames(test) %in% "day"] < -"given"

whoref5 < -plyr::rbind.fill(whoref5,test)

}

whoref5 < -whoref5%>%

mutate(indicator=case\_when(!indicator %in% "lhfa" ~ indicator,

indicator %in% "lhfa" ~ "hfa"))%>% mutate(given=case\_when(!indicator %in% c("wfh","wfl")~ given,

indicator %in% "wfh" ~ height,

indicator %in% "wfl" ~ length))

whoref5 < -whoref5[,!names(whoref5) %in% c("height", "length")]

whoref5 < -as.data.frame(whoref5)

str(whoref5)

*# study data <= 5 years old*

dat5 < -dat[dat$age.m < =60,]

## 'data.frame': 13546 obs. of 15 variables:

## $ indicator: chr "bfa" "bfa" "bfa" "bfa" ...

## $ given : num 0 1 2 3 4 5 6 7 8 9 ...

## $ l : num -0.3053 -0.1867 -0.0681 0.0505 0.169 ...

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | $ m : | | | num | 13.4 13.4 13.4 13.4 | | | | 13.4 ... | |
| ## | $ s : | | | num | 0.0956 0.096 0.0963 | | | | 0.0967 0.0971 ... | |
| ## | $ sd4neg : | | | num | 9.23 9.15 9.06 8.96 | | | | 8.86 ... | |
| ## | $ | sd3neg | : | num | 10.18 10.12 10.06 9.99 9.92 ... | | | | | |
| ## | $ | sd2neg | : | num | 11.1 11.1 11.1 11 11 ... | | | | | |
| ## | $ sd1neg : | | | num | 12.2 | 12.2 | 12.2 | 12.1 | 12.1 | ... |
| ## | $ sd0 : | | | num | 13.4 | 13.4 | 13.4 | 13.4 | 13.4 | ... |
| ## | $ sd1 : | | | num | 14.8 | 14.8 | 14.7 | 14.7 | 14.7 | ... |
| ## | $ sd2 : | | | num | 16.3 | 16.3 | 16.3 | 16.2 | 16.2 | ... |
| ## | $ sd3 : | | | num | 18.1 | 18 17.9 17.8 17.8 ... | | | | |
| ## | $ sd4 : | | | num | 19.9 | 19.7 19.6 19.5 19.3 ... | | | | |
| ## | $ | sex | : | num | 1 1 1 1 1 1 1 1 1 1 ... | | | | | |

#### Write new function for z-score calculation

Second, we write a function for height, weight, BMI for age and weight for height z-score calculation. Note that age should be in month for children >5 years old and age should be in days for children ≤ 5 years old with no decimal digit. Height should be in cm with 1 decimal digit.

To simplify, height will be used for all index calculation: Height = supine length−0.7 cm (if standing = 2 = supine). If standing status is unknown and the child is < 731 days old so the measure of the child is regarded as supine length. If standing status is unknown and the child is ≥ 731 days old, so the measure of the child is regarded as standing height.

Our new function *zcal()* below is designed to:

* Accept a study data frame (“*data*”), the correct reference table (“*ref.dat*”), and column names.
* Correct for supine length vs. standing height.
* Calculate BMI from the (potentially corrected) height.
* Join the study data to the WHO reference table by sex and age (either days or months).
* Calculate the Z-scores for height for age (HFA), weight for age (WFA), BMI for age (BFA), and weight for height (WFH) using the official LMS formula.
* Handle extreme Z-scores as the WHO recommends (by extrapolating from the SD bands).

zcal < -function(data=dat5,

ref.dat=whoref5,

sex.numeric= "sex\_numeric", *#sex numeric 1=male, 2= female* age.month=NULL, *#age in months, for age >5 years old* age.day=NULL, *#age in days, for age 0-5 years old* length.height.cm="height", *# length or height in cm* weight.kg="weight", *# weight in kg* index=c("hfa","wfa","bfa","wfh"), *# calculate these indices* standing.var=NULL, *#1=standing, 2= supine, 3=unknown*

age.group=c(" < =5 years",">5 years")){

*# Calculate age in months or age in days # if required but not available*

if (age.group==" < =5 years" & is.null(age.day)){

data$age.day < -round(data$age.month\*30.4,0)

}

if (age.group==">5 years" & is.null(age.month)){ data$age.month < -round(data$age.day/30.4,0)

}

if (!is.null(age.month)){

data$age.month < -round(data[,age.month],0)

}

if (!is.null(age.day)){

data$age.day < -round(data[,age.day],0)

}

data$sex.numeric < -data[,sex.numeric]

data$length.height.cm < -round(data[,length.height.cm],1) data$weight.kg < -data[,weight.kg]

data$standing.var < -data[,standing.var]

*# corrected height (heightc) corresponding to standing status*

if (age.group==">5 years"){ data < -data %>%

mutate(heightc=length.height.cm)

}

if (age.group==" < =5 years" & is.null(standing.var)){

data < -data %>%

mutate(heightc=case\_when(age.day < 731 ~ length.height.cm -0.7,

age.day>=731 ~ length.height.cm))

}

if (age.group==" < =5 years" & !is.null(standing.var)){

data < -data %>%

mutate(heightc=case\_when(standing.var==1 ~

length.height.cm, standing.var==2 ~

length.height.cm-0.7,

standing.var==3 & age.day < 731 ~

length.height.cm -0.7, standing.var==3 & age.day>=731 ~

length.height.cm))

}

data < -data %>% mutate(bmi=(weight.kg/((heightc/100)ˆ2)))

*#Height*

if (age.group==">5 years"){

datc < -left\_join(data,subset(ref.dat,indicator %in% "hfa"),

by = c(age.month = "given", sex.numeric="sex"))

}

if (age.group==" < =5 years"){

datc < -left\_join(data,subset(ref.dat,indicator %in% "hfa"),

by = c(age.day = "given", sex.numeric="sex"))

}

datc < -datc %>%

mutate(sd23pos = sd3 - sd2,

sd23neg = sd2neg - sd3neg)%>%

mutate(hfaz = (((heightc/m)ˆl) - 1)/(l \*s)) %>%

*#manage extreme z values*

mutate(hfaz =case\_when(hfaz >= -3 & hfaz < =3 ~ hfaz,

hfaz>3 ~ 3 + ((heightc - sd3)/sd23pos),

hfaz < -3 ~ ((heightc - sd3neg)/sd23neg)))

datz < - datc[,!colnames(datc) %in% c(colnames(ref.dat),

"sd23pos","sd23neg")]

*#Weight*

if (age.group==">5 years"){

datc < -left\_join(datz,subset(ref.dat,indicator %in% "wfa"),

by = c(age.month = "given", sex.numeric="sex"))

}

if (age.group==" < =5 years"){

datc < -left\_join(datz,subset(ref.dat,indicator %in% "wfa"),

by = c(age.day = "given", sex.numeric="sex"))

}

datc < -datc %>%

mutate(sd23pos = sd3 - sd2,

sd23neg = sd2neg - sd3neg)%>%

mutate(wfaz = (((weight.kg/m)ˆl) - 1)/(l \*s)) %>%

*#manage extreme z values*

mutate(wfaz =case\_when(wfaz >= -3 & wfaz < =3 ~ wfaz,

wfaz>3 ~ 3 + ((weight.kg - sd3)/sd23pos),

wfaz < -3 ~ ((weight.kg - sd3neg)/sd23neg)))

datz < - datc[,!colnames(datc) %in% c(colnames(ref.dat),

"sd23pos","sd23neg")]

*#BMI*

if (age.group==">5 years"){

datc < -left\_join(datz,subset(ref.dat,indicator %in% "bfa"),

by = c(age.month = "given", sex.numeric="sex"))

}

if (age.group==" < =5 years"){

datc < -left\_join(datz,subset(ref.dat,indicator %in% "bfa"),

by = c(age.day = "given", sex.numeric="sex"))

}

datc < -datc %>%

mutate(sd23pos = sd3 - sd2,

sd23neg = sd2neg - sd3neg)%>%

mutate(bfaz = (((bmi/m)ˆl) - 1)/(l \*s)) %>% *#manage extreme z values*

mutate(bfaz =case\_when(bfaz >= -3 & bfaz < =3 ~ bfaz,

bfaz>3 ~ 3 + ((bmi - sd3)/sd23pos),

bfaz < -3 ~ ((bmi - sd3neg)/sd23neg)))

datz < - datc[,!colnames(datc) %in% c(colnames(ref.dat),

"sd23pos","sd23neg")]

*#WFH*

if (age.group==">5 years"){ datz$wfhz < -NA

}

if (age.group==" < =5 years"){

datc < -left\_join(datz, subset(ref.dat, indicator %in% "wfh"),

by = c(heightc = "given", sex.numeric="sex"))

datc < -datc %>% mutate(sd23pos = sd3 - sd2,

sd23neg = sd2neg - sd3neg)%>%

mutate(wfhz = (((weight.kg/m)ˆl) - 1)/(l \*s)) %>%

*#manage extreme z values*

mutate(wfhz =case\_when(wfhz >= -3 & wfhz < =3 ~ wfhz,

wfhz>3 ~ 3 + ((weight.kg - sd3)/sd23pos),

wfhz < -3 ~ ((weight.kg - sd3neg)/sd23neg)))

datz < - datc[,!colnames(datc) %in% c(colnames(ref.dat),

"sd23pos","sd23neg")]

}

return(datz)

}

#### Apply the new function for z-score calculation

Third, we apply the function *zcal()* written above to calculate and add Z-scores to study dataset of children. Below is an example application for dataset of children > 5 years old.

datz18 < -zcal(data=dat18,

ref.dat=whoref19,

sex.numeric= "sex\_numeric", age.month="age.m",

age.day="age.d", length.height.cm="height", weight.kg="weight",

standing.var=NULL,

age.group=">5 years")

str(datz18)

## 'data.frame': 79224 obs. of 25 variables:

## $ cid : Factor w/ 88678 levels "c1","c2","c3",..: 1 21 21 22 27 34 37 39 46 53 ...

## $ weight : num 33.1 56.4 65 77 60.7 77.8 44.6 82.9 41.2 52.3 ...

## $ height : num 135 154 157 166 171 ...

## $ bmi : num 18.2 23.8 26.2 28.1 20.8 ...

## $ site : Factor w/ 3 levels "Site1","Site2",..: 2 2 2 2 2 2 2 2 2 2 ...

## $ age : num 8.6 12.9 14 16.5 15.6 13.9 15 17.6 10.3 9.9 ...

## $ age.d : num 3124 4726 5122 6026 5706 ...

## $ age.m : num 103 155 168 198 187 167 180 212 123 119 ...

## $ age.y : Factor w/ 18 levels "2","3","4","5",..: 8 12 13 15 15 13 14 17 9 9 ...

## $ yearvisit : num 1 0 1 1 1 1 1 1 1 1 ...

## $ year : Factor w/ 6 levels "0","1","2","4",..: 2 1 2 2 2 2 2 2 2 2 ...

## $ age.cat : Factor w/ 3 levels "age 0-5 years",..: 2 3 3 3 3 3 3 3 2 2 ...

## $ bmi.cat : Factor w/ 8 levels "BMI<=10","10<BMI <=15",..: 3 4 5 5 4 4 3 5 3 4 ...

## $ sex\_numeric : num 2 1 1 1 1 1 2 1 2 1 ...

## $ age.month : num 103 155 168 198 187 167 180 212 123 119 ...

## $ age.day : num 3124 4726 5122 6026 5706 ...

## $ sex.numeric : num 2 1 1 1 1 1 2 1 2 1 ...

## $ length.height.cm: num 135 154 157 166 171 ...

## $ weight.kg : num 33.1 56.4 65 77 60.7 77.8 44.6 82.9 41.2 52.3 ...

## $ standing.var :'data.frame': 79224 obs. of 0 variables

## $ heightc : num 135 154 157 166 171 ...

## $ hfaz : num 0.8369 -0.1936 -0.7516 -1.1178 -0.0985 ...

## $ wfaz : num 1.2 NA NA NA NA ...

## $ bfaz : num 1.04 1.81 2.06 1.96 0.24 ...

## $ wfhz : logi NA NA NA NA NA NA ...

Also, the below example applies the function *zcal()* written above to calculate and add Z-scores to study dataset of children ≤ 5 years old.

datz5 < -zcal(data=dat5,

ref.dat=whoref5,

sex.numeric= "sex\_numeric", age.month="age.m",

age.day="age.d", length.height.cm="height", weight.kg="weight",

standing.var=NULL,

age.group=" < =5 years")

str(datz5)

## 'data.frame': 18113 obs. of 25 variables:

## $ cid : Factor w/ 88678 levels "c1","c2","c3",..: 828 1181 1230 1684 1726 1726

2132 2694 2845 3077 ...

## $ weight : num 22 17.5 18 20.2 20.5 18.2 21.9 16.5 16.8 17.7 ...

## $ height : num 112 108 106 104 106 ...

## $ bmi : num 17.5 15.1 16 18.7 18.2 ...

## $ site : Factor w/ 3 levels "Site1","Site2",..: 2 2 2 2 2 2 2 2 2 2 ...

## $ age : num 4.9 5 4.8 4.8 4.9 4.4 4.9 5 4.6 4.7 ...

## $ age.d : num 1807 1827 1750 1758 1789 ...

## $ age.m : num 59 60 57 58 59 53 58 60 55 56 ...

## $ age.y : Factor w/ 18 levels "2","3","4","5",..: 4 4 4 4 4 3 4 4 4 4 ...

## $ yearvisit : num 0 0 0 0 0 0 0 1 0 0 ...

## $ year : Factor w/ 6 levels "0","1","2","4",..: 1 1 1 1 1 1 1 2 1 1 ...

## $ age.cat : Factor w/ 3 levels "age 0-5 years",..: 1 1 1 1 1 1 1 1 1 1 ...

## $ bmi.cat : Factor w/ 8 levels "BMI<=10","10<BMI <=15",..: 3 3 3 3 3 3 3 2 3 3 ...

## $ sex\_numeric : num 1 1 2 2 2 2 2 1 1 2 ...

## $ age.month : num 59 60 57 58 59 53 58 60 55 56 ...

## $ age.day : num 1807 1827 1750 1758 1789 ...

## $ sex.numeric : num 1 1 2 2 2 2 2 1 1 2 ...

## $ length.height.cm: num 112 108 106 104 106 ...

## $ weight.kg : num 22 17.5 18 20.2 20.5 18.2 21.9 16.5 16.8 17.7 ...

## $ standing.var :'data.frame': 18113 obs. of 0 variables ## $ heightc : num 112 108 106 104 106 ...

## $ hfaz : num 0.517 -0.535 -0.448 -0.905 -0.589 ...

## $ wfaz : num 1.3744 -0.3483 0.0841 0.8347 0.8611 ...

## $ bfaz : num 1.5754 -0.0365 0.4927 1.965 1.7386 ...

## $ wfhz : num 1.464 -0.102 0.534 2.145 1.875 ...

The above *zcal()* function may also be applied to calculate Z-scores for individual. The below example calculates Z-scores for a female 71 months old with weight = 19 kg, and height = 115 cm. We first need to create a data frame for an individual then apply the function to the data frame.

dati < -as.data.frame(cbind(sex.numeric=2,age.month=71,

height.cm=115,weight.kg=19))

datzi < -zcal(data=dati,

ref.dat=whoref19,

sex.numeric="sex.numeric", age.month="age.month",

age.day=NULL,

length.height.cm="height.cm", weight.kg="weight.kg",

age.group=">5 years",

standing.var=NULL)

datzi[,7:12]

## heightc bmi hfaz wfaz bfaz wfhz

## 1 115 14.4 0.0703 -0.347 -0.617 NA

The example below calculates z-scores for a male 31 months old with weight = 11 kg, height = 105 cm, supine.

dati < -as.data.frame(cbind(sex.numeric=1,

age.month=31,

length.cm=105,

weight.kg=11, standing=2))

datzi < -zcal(data=dati,

ref.dat=whoref5,

sex.numeric="sex.numeric", age.month="age.month",

age.day=NULL,

length.height.cm="length.cm", weight.kg="weight.kg",

age.group=" < =5 years", standing.var="standing")

datzi[,9:14]

## heightc bmi hfaz wfaz bfaz wfhz ## 1 104 10.1 3.38 -1.72 -2.6 -1.91

### Calculate Prevalence from z-Scores Manually

Now that we have Z-scores for anthropometric measures, we can do nutritional status classification following WHO references and calculate their prevalence for our study data. Below are some examples for these calculations for the two age groups.

#### For Age ≤ 5 Years Old

According to WHO, for children ≤ 5 years old, weight-for-height (WFH) is used for obesity, wasting classification. The below example classifies WFH Z-scores into categories “Obesity”, “Overweight”, “Normal WFH”, “Wasting”, calculates the prevalence and 95%CI of these categories and displays the results as a table.

*#w5l < -c("Obesity","Overweight","Normal WFH","Wasting")*

datz5 < -datz5 %>% mutate(wfh.catw= cut(

wfhz,

breaks = c(-Inf,-2, 2, 3, Inf), *# Define break points*

labels = c("Wasting", "Normal WFH", "Overweight","Obesity"), *# Assign labels*

right = TRUE))

datz5$wfh.catw < -factor(datz5$wfh.catw, levels=w5l) tbl < - table(datz5$wfh.catw)

pci < -round(MultinomCI(tbl,

conf.level=0.95, method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep=""))

kable(tblb,

caption="Prevalence, 95\\%CI Obesity, Wasting calculated manually")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 28**. Prevalence, 95%CI Obesity, Wasting calculated manually.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Obesity | 410 | 0.02314 (0.0193, 0.02705) |
| Overweight | 905 | 0.05107 (0.04723, 0.05498) |
| Normal WFH | 16,201 | 0.91423 (0.91039, 0.91814) |
| Wasting | 205 | 0.01157 (0.00773, 0.01548) |

Source: Table by author(s).

Similarly, the example below classifies heigh-for-age (HFA) z-cores into categories “Extreme tallness”, “Normal HFA”, and “Stunting”, calculates the prevalence and 95%CI of these categories and displays results as a table.

*#stl < -c("Extreme tallness", "Normal HFA", "Stunting")*

datz5 < -datz5 %>% mutate(height.cats= cut(

hfaz,

breaks = c(-Inf,-2, 3, Inf), *# Define break points*

labels = c("Stunting", "Normal HFA", "Extreme tallness"), *# labels*

right = TRUE

))

datz5$height.cats < -factor(datz5$height.cats, levels=stl)

tbl < - table(datz5$height.cats)

pci < -round(MultinomCI(tbl,

conf.level=0.95,

method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep=""))

kable(tblb,

caption="Prevalence, 95\\%CI stunting under 5 calculated manually")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 29**. Prevalence, 95%CI stunting under 5 calculated manually.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Extreme tallness | 48 | 0.00267 (0.00267, 0.00277) |
| Normal HFA | 17,367 | 0.96446 (0.96446, 0.96457) |
| Stunting | 592 | 0.03288 (0.03288, 0.03298) |

Source: Table by author(s).

#### For Age > 5 Years Old

For children >5 years old, BMI-for-age is used for obesity, thinness classification. The below example calculates the prevalence and 95%CI of “Obesity”, “Overweight”, “Normal BMI”, “Thinness” and displays results as a table similarly to the above.

*#t18l < -c("Obesity","Overweight","Normal BMI","Thinness")*

datz18 < -datz18 %>% mutate(bmi.catt= cut(

bfaz,

breaks = c(-Inf,-2, 1, 2, Inf), *# Define break points*

labels = c("Thinness", "Normal BMI", "Overweight","Obesity"), *# labels*

right = TRUE

))

datz18$bmi.catt < -factor(datz18$bmi.catt, levels=t18l)

tbl < - table(datz18$bmi.catt)

pci < -round(MultinomCI(tbl,

conf.level=0.95,

method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1],

" (",

pci[,2],

", ",

pci[,3],

")",

sep=""))

kable(tblb,

caption="Prevalence, 95\\%CI Obesity, Thinness calculated manually")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 30**. Prevalence, 95%CI Obesity, Thinness calculated manually.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Obesity | 11,992 | 0.15253 (0.14896, 0.1561) |
| Overweight | 17,925 | 0.22799 (0.22443, 0.23156) |
| Normal BMI | 46,752 | 0.59464 (0.59107, 0.59821) |
| Thinness | 1954 | 0.02485 (0.02129, 0.02843) |

Source: Table by author(s).

Again, similarly, the below example calculates the prevalence and 95%CI of “Extreme tallness”, “Normal HFA”, “Stunting” and displays results as a table.

*#stl < -c("Extreme tallness", "Normal HFA", "Stunting")*

datz18 < -datz18 %>% mutate(height.cats= cut(

hfaz,

breaks = c(-Inf,-2, 3, Inf), *# Define break points*

labels = c("Stunting", "Normal HFA", "Extreme tallness"), *# labels*

right = TRUE

))

datz18$height.cats < -factor(datz18$height.cats, levels=stl)

tbl < - table(datz18$height.cats)

pci < -round(MultinomCI(tbl,

conf.level=0.95,

method="sisonglaz"),5)

tblb < - cbind(count=tbl, p.95ci=paste(pci[,1], " (",

pci[,2], ", ",

pci[,3], ")",

sep=""))

kable(tblb,

caption="Prevalence, 95\\%CI of stunting calculated manually")%>% kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 31**. Prevalence, 95%CI of stunting calculated manually.

|  |  |  |
| --- | --- | --- |
|  | **Count** | **p.95ci** |
| Extreme tallness | 241 | 0.00305 (0.00304, 0.00309) |
| Normal HFA | 78,121 | 0.98974 (0.98973, 0.98977) |
| Stunting | 569 | 0.00721 (0.0072, 0.00724) |

Source: Table by author(s).

### WHO LMS Method for Constructing Normalized Growth Curves

The LMS (Lambda, Mu and Sigma) method enables efficient calculation of percentiles and Z-scores and smoothing of growth curves [80] and may be done with “*gamlss”* package [74]. The example below fits the LMS model for height for boys ≤ 5 years old: *lms()* from the “*gamlss”* package tests several distributions (*families=c("BCCGo","BCPEo","BCTo")*) and select the best family to find the best-fitting L, M, and S parameters for height as a smooth function of “*age.m*” (*trans.x = TRUE* : Box–Cox transformation for age). The selected model is the one with lowest deviance.

dat05b.h < -na.omit(dat05b[, c("height","age.m")])

lms05b.h < -lms(height,age.m,families=c("BCCGo","BCPEo","BCTo"), data=dat05b.h,points=FALSE,legend=TRUE,

k=3,calibration=F, trans.x=T)

## \*\*\* Checking for transformation for x \*\*\*

## \*\*\* power parameters 0.64 \*\*\*

## \*\*\* Initial fit\*\*\*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | GAMLSS-RS iteration 1: | Global | Deviance | = | 55128 |
| ## | GAMLSS-RS iteration 2: | Global | Deviance | = | 55128 |
| ## | \*\*\* Fitting BCCGo \*\*\* |  |  |  |  |
| ## | GAMLSS-RS iteration 1: | Global | Deviance | = | 54049 |
| ## | GAMLSS-RS iteration 2: | Global | Deviance | = | 53993 |
| ## | GAMLSS-RS iteration 3: | Global | Deviance | = | 53993 |
| ## | GAMLSS-RS iteration 4: | Global | Deviance | = | 53993 |
| ## | \*\*\* Fitting BCPEo \*\*\* |  |  |  |  |
| ## | GAMLSS-RS iteration 1: | Global | Deviance | = | 54134 |
| ## | GAMLSS-RS iteration 2: | Global | Deviance | = | 53837 |
| ## | GAMLSS-RS iteration 3: | Global | Deviance | = | 53837 |
| ## | GAMLSS-RS iteration 4: | Global | Deviance | = | 53837 |
| ## | GAMLSS-RS iteration 5: | Global | Deviance | = | 53837 |
| ## | \*\*\* Fitting BCTo \*\*\* |  |  |  |  |
| ## | GAMLSS-RS iteration 1: | Global | Deviance | = | 53751 |
| ## | GAMLSS-RS iteration 2: | Global | Deviance | = | 53710 |
| ## | GAMLSS-RS iteration 3: | Global | Deviance | = | 53709 |
| ## | GAMLSS-RS iteration 4: | Global | Deviance | = | 53709 |
| ## | GAMLSS-RS iteration 5: | Global | Deviance | = | 53709 |
| ## | GAMLSS-RS iteration 6: | Global | Deviance | = | 53709 |

## % of cases below 0.4 centile is 0.313

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | % | of | cases | below | 2 centile is | | | 1.92 |
| ## | % | of | cases | below | 10 | centile | is | 10.6 |
| ## | % | of | cases | below | 25 | centile | is | 25.6 |
| ## | % | of | cases | below | 50 | centile | is | 49.9 |
| ## | % | of | cases | below | 75 | centile | is | 74.7 |
| ## | % | of | cases | below | 90 | centile | is | 89.8 |
| ## | % | of | cases | below | 98 | centile | is | 98.4 |
| ## | % | of | cases | below | 99.6 centile is 99.7 | | | |

lms05b.h$family ## [1] "BCTo" "Box-Cox-t-orig."

lms05b.h$power

## [1] 0.64

Next, we construct nice looking growth curves for height-for-age for boys ≤ 5 years old: *centiles()* of the “*gamlss”* package takes our fitted LMS model above (“*lms05b.h*”) and uses it to draw the smoothed percentile curves (0.4th, 2nd, 50th, 98th,...) across the range of ages.

centiles(lms05b.h, dat05b.h$age.m,

cent=c(0.4,2,10,25,50,75,90,98,99.6),

ylab="Height (cm)", xlab="Age (month)",

legend=FALSE,

ylim=c(70,130),

main="Centile curves height for age (boys, 18 to 60 months)",

yaxt = "n", xaxt = "n",

points=FALSE, save = FALSE)

|  |  |  |
| --- | --- | --- |
| axis(1, | at = seq(16, | 60, by = 2), las=2) |
| axis(2, | at = seq(70, | 130, by = 5), las=2) |

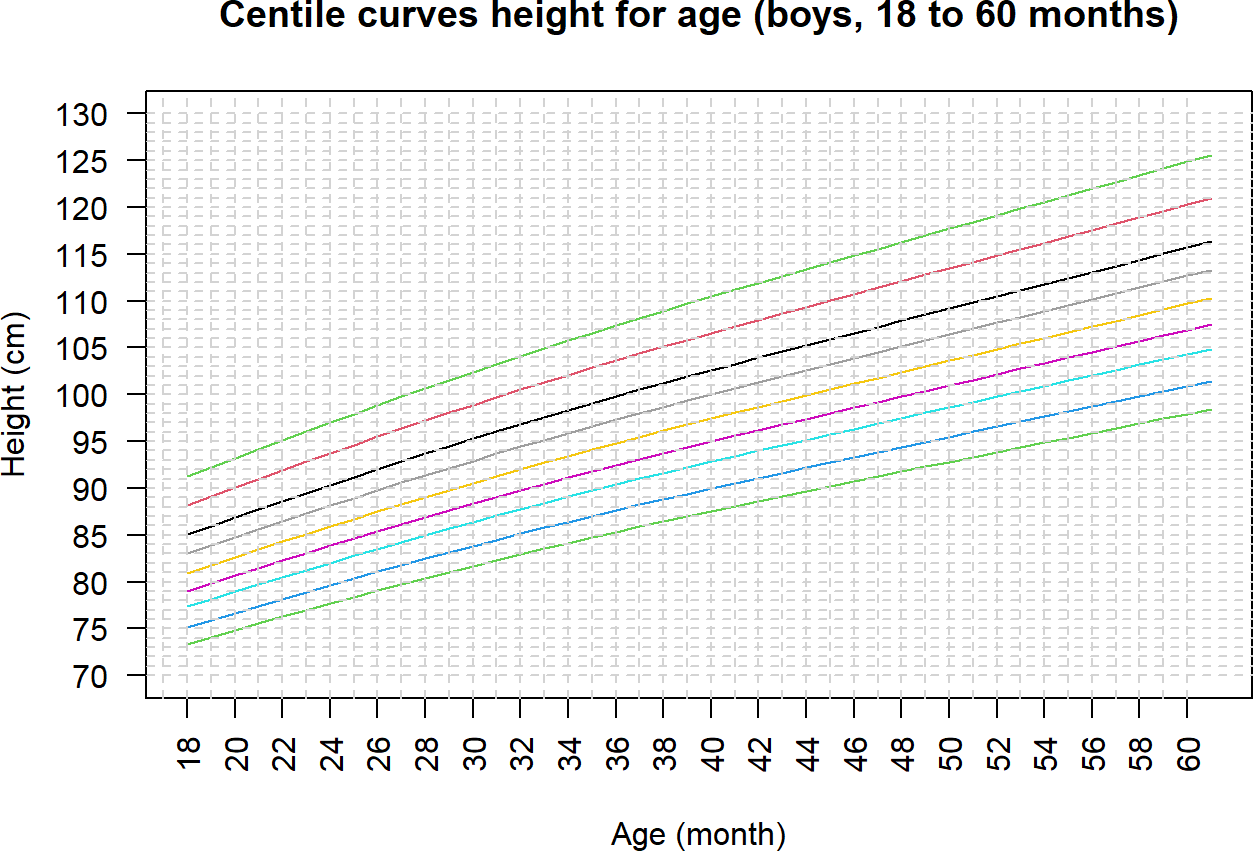
*#Add horizontal grid*

abline(h = seq(70, 130, by = 1), lty = 2, col = "lightgrey")

*#Add vertical grid*

abline(v = seq(16, 60, by = 1), lty = 2, col = "lightgrey")





Source: Figure by author(s).

**Figure 107**. Centile curves height for age.

We can also create a fan chart using *centiles.fan()* of the “*gamlss”* package which produces shaded percentile bands and provides a visually intuitive growth distribution.

centiles.fan(lms05b.h,dat05b.h$age.m,

cent=c(0.4,2,10,25,50,75,90,98,99.6),

ylab="Height (cm)", xlab="Age (month)", ylim=c(70,130),

yaxt = "n", xaxt = "n",

main="Fan-chart height for age (boys, 18 to 60 months)",

points=FALSE)

axis(1, at = seq(16, 60, by = 2), las=2)

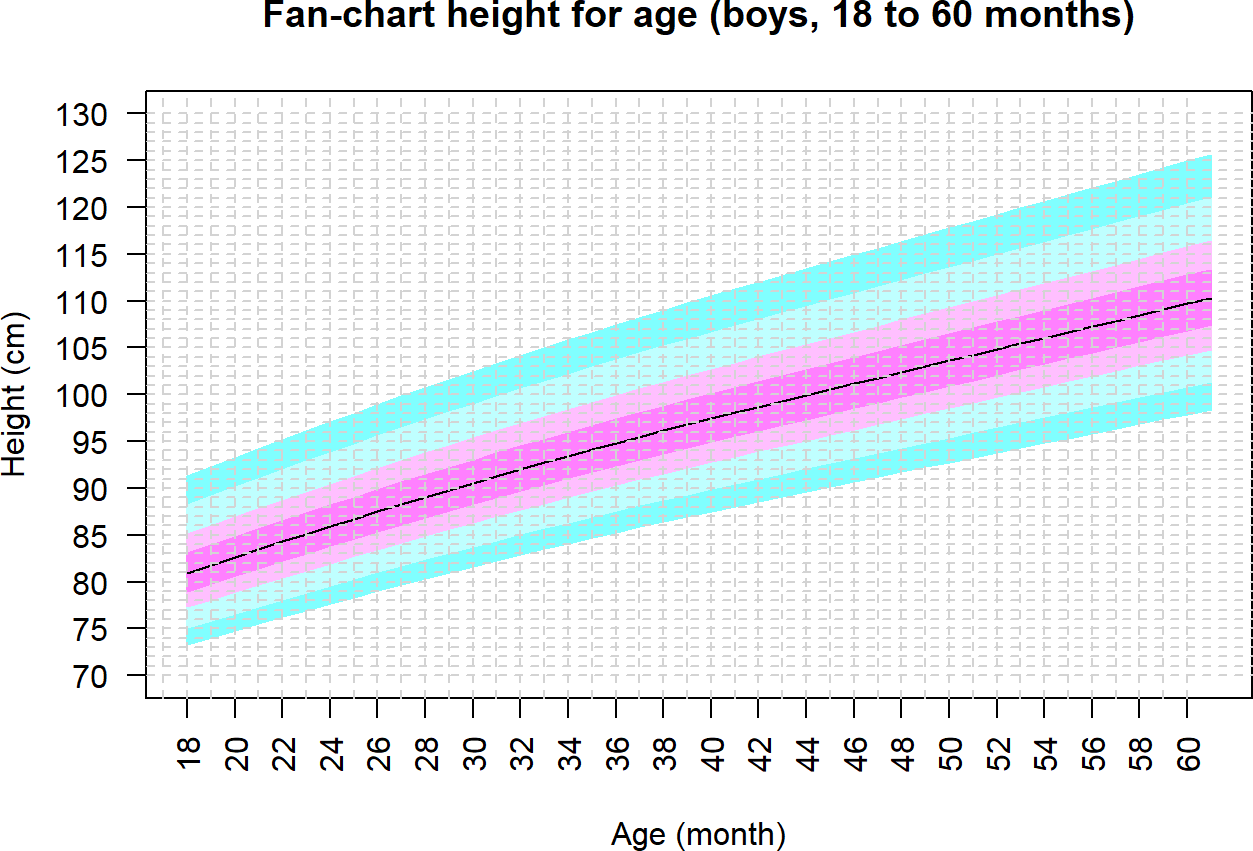
axis(2, at = seq(70, 130, by = 5), las=2)

*#Add horizontal grid*

abline(h = seq(70, 130, by = 1), lty = 2, col = "lightgrey")

*#Add vertical grid*

abline(v = seq(16, 60, by = 1), lty = 2, col = "lightgrey")



Source: Figure by author(s).

**Figure 108**. Fan-chart height for age.

Above are example R codes for some basic analysis of child physical growth data. One may adapt the codes and elaborate the analysis for their own data.

## Useful Resources

WHO 2007 child physical growth reference data may be downloaded from WHO website for for [0 to 5 years old](https://www.who.int/tools/child-growth-standards/standards) and [5 to 19 years old](https://www.who.int/tools/growth-reference-data-for-5to19-years/) [68], [69].

The original paper “Overweight & obesity epidemic, temporal trends and regional disparities in physical growth of Vietnamese children” published by Nhan Thi Ho et al [81].

## Chapter Summary

* Newly created functions *iqr()* for median, quantile range plotting and *medci()*

for median, 95%CI plotting with ggplot2

* Function *MedianCI()* of the *DescTools* for median, 95%CI of anthropometric measures
* Function *ci\_median\_diff()* of the *confintr* package for difference between medians and 95%CI of the difference
* Average temporal trend of data with repeated measures may be evaluated with GEE model
* Two sets of WHO 2007 reference data (≤ 5 years old and >5 to 19 years old)
* Color code and ggplot2 for visualize observed data vs. (WHO) references
* *“anthro”* and “*zscorer”* package to calculate Z-scores for ≤ 5 years old
* Newly created function *zcal()* to calculate Z-scores for ≤ 5 years old and > 5 to 19 years old for dataset or individual
* Nutritional status classification (obesity, wasting, stunting,. . . ) based on Z-scores following WHO recommendation
* Function *MultinomCI()* of the “*DescTools”* package for prevalence, 95%CI of nutritional status using Glaz & Sison’s method
* *lms()* function of the *gamlss* package for construction of growth curves using WHO LMS (Lambda, Mu and Sigma) method

# Analysis of Clinical Trial Data with R

This chapter provides practical guide to many steps for clinical trials using R. It includes example R codes for clinical trial planning including sample size – power estimation for superiority and non-inferiority trials for most common types of outcomes as well as for randomization. With a detailed case study based on an example data mimicking a vaccine trial, this chapter demonstrates how to use some specialized R packages to generate baseline summaries, evaluate efficacy outcomes, and summarize adverse events. Its hands-on examples are designed to help bio-medical researchers effectively work with clinical trial data to generate reporting-ready tables and figures.

## Why R for Clinical Trial Data Analysis?

While traditional software like SAS has a long history in clinical trial analysis, R offers several distinct advantages such as open-source and free, vast package ecosystem, superior graphics, dynamic and reproducible reports, and growing active community, contribution. These advantages are driving its rapidly increasing adoption in clinical trial analysis, particularly in academia, clinical and bio-medical research.

First, we load the required packages for clinical trial data analysis:

* *"pwr" [24],"powerSurvEpi" [82],"TrialSize" [83], "SampleSize4ClinicalTrials" [84], "gsDesign"* [85] for sample size – power estimation.
* *"blockrand" [86], “randomizeR”* [87] for randomization

Other packages we will load later for our example clinical trial data analysis:

* *“arsenal”* [19] for baseline summary
* “*survival*” [43] for analysis of time-to-event outcomes (e.g. for efficacy analysis)
* “*tidyverse*” [11] for data manipulation and summary (e.g. for adverse event summary)

Other supporting packages ("*rio*" [14], "*knitr*" [5], "*kableExtra*" [12]) are routinely use as in previous chapters.

*#load multiple packages*

packages < -c("rio","tidyverse","knitr", "kableExtra", "pwr","powerSurvEpi","TrialSize", "SampleSize4ClinicalTrials", "gsDesign","blockrand")

lapply(packages, library, character.only = TRUE)

## Sample Size—Power Estimation for Clinical Trial

Sample size estimation is a fundamental component of clinical trial design, ensuring the study is appropriately powered to detect a clinically meaningful effect if one truly exists. Below are examples for sample size estimation for various clinical trial designs and outcomes [88].

### Superiority Trials

Superiority trials are designed to show that the new treatment is better than the control. We go through examples for estimating sample size for the three most common types of outcomes.

#### Continuous Outcome

In our example below, we are designing a trial to test if a new drug reduces LDL cholesterol more than a placebo. We want to detect a mean difference of 10 mg/dL. Based on prior research, we expect the standard deviation in both groups to be 20 mg/dL. We desire 90% power with a two-sided alpha of 0.05. Package “*pwr”* [24] may be used. The effect size “*d*” is calculated as the mean difference divided by the standard deviation (10/20). The *pwr.t.test()* function takes this standardized effect size, the desired power (0.90), and significance level (0.05) to solve for sample size “*n*”.

*#library(pwr)*

*# Parameters*

mu\_diff < - 10 *# Expected difference in means*

sd < - 20 *# Expected standard deviation*

sig\_level < - 0.05 *# Significance level (alpha)*

power < - 0.90 *# Desired power (1 - beta)*

*# Calculate Cohen's d (standardized effect size)*

cohens\_d < - mu\_diff / sd

*# Perform the sample size calculation*

ss\_continuous < - pwr.t.test(

d = cohens\_d,

power = power,

sig.level = sig\_level,

type = "two.sample",

alternative = "two.sided"

)

ss\_continuous

##

## Two-sample t test power calculation

##

## n = 85

## d = 0.5

## sig.level = 0.05

## power = 0.9

## alternative = two.sided

##

## NOTE: n is number in \*each\* group

The output shows that 85 patients needed per group (a total of 170 participants).

#### Binary Outcome

For our example below, a new cancer therapy is expected to have a 50% response rate, compared to a 35% response rate for the standard therapy. We want to determine the sample size needed to detect this difference with 80% power and a two-sided alpha of 0.05. We first calculate effect size Cohen's h using the function *ES.h()* then calculate sample size using the function *pwr.2p.test()*.

*#library(pwr)*

*# Parameters*

p1 < - 0.50 *# Expected proportion in the new therapy group*

p2 < - 0.35 *# Expected proportion in the standard therapy group*

sig\_level < - 0.05

power < - 0.80

*# Calculate effect size using Cohen's h*

effect\_size\_h < - ES.h(p1 = p1, p2 = p2)

*# Perform the sample size calculation*

ss\_binary < - pwr.2p.test(

h = effect\_size\_h,

power = power,

sig.level = sig\_level,

alternative = "two.sided"

)

ss\_binary

##

## Difference of proportion power calculation for binomial distribution (arcsine transformation)

##

## h = 0.305

## n = 169

## sig.level = 0.05

## power = 0.8

## alternative = two.sided

##

## NOTE: same sample sizes

The output shows that 169 patients are needed per group (a total of 352 participants).

#### Time-to-Event Outcome

Example: A clinical trial is planned to test a new drug for advanced pancreatic cancer. Ratio of participants in experimental group (E) compared to control group

(C) is 1. Probability of death in experimental group over the maximum time period of the study is 0.35. Probability of death in control group over the maximum time period of the study is 0.5. Postulated hazard ratio is 0.7. The desired power is 80% with a two-sided alpha of 0.05. The package “*powerSurvEpi”* [82] may be used.

*#library(powerSurvEpi)*

ss\_survival < -ssizeCT.default(power = 0.8,

k = 1,

pE = 0.35,

pC = 0.5,

RR = 0.7,

alpha = 0.05)

ss\_survival

## nE nC ## 297 297

The output shows that the sample size needed for each group is 297 (a total of 594 participants).

### Non-Inferiority Trial

Non-inferiority trial is designed to show that a new treatment is not worse than an active control.

#### Continuous Outcome

Example: A new, cheaper anti-hypertensive drug is being tested against the standard drug. The standard drug reduces systolic blood pressure (SBP) by 25 mmHg. The non-inferiority margin (∆) is set at 5 mmHg. We assume pooled standard deviation of SBP of two groups is 15 mmHg, the true mean difference in SBP between two groups is 0. We want 90% power with a one-sided alpha of 0.025. The function *TwoSampleMean.NIS()* of the “*TrialSize”* package [83] may be used. This function takes our defined values for one-sided significance level (alpha), type 2 error (beta =1 -desired power), assumed true difference (margin), assumed SD of the outcome (sigma), the non-inferiority margin (delta), and ratio of n\_new / n\_standard (k) to estimate the required sample size n for our study.

*#library(TrialSize)*

*# Define Parameters*

alpha < - 0.025

power < - 0.90

true\_difference < - 0

sd\_outcome < - 15 ni\_margin < - 5 allocation\_ratio < - 1

*# One-sided significance level*

*# Desired power (so beta is 0.10)*

*# Assumed true mu\_new - mu\_standard*

*# Assumed SD of the outcome*

*# The non-inferiority margin (*Δ*)*

*# k: ratio of n\_new / n\_standard*

*# Perform the Sample Size Calculation*

ss\_ni\_continuous < - TwoSampleMean.NIS(

alpha = alpha,

beta = 1 - power,

sigma = sd\_outcome,

k = allocation\_ratio,

delta = ni\_margin,

margin = true\_difference

)

ss\_ni\_continuous

## [1] 189

The output shows the sample size needed for each group (189 participants).

#### Binary Outcome

Example: A trial comparing a new single-dose antibiotic to a standard 7-day course. We assume the true success rate of the new antibiotic (p1) and the success rate of the standard treatment (p2) is 90%. The non-inferiority margin is 10%. We want 80% power with a one-sided alpha of 0.025. We use the function *TwoSampleProportion.NIS()* and input the parameters similarly to the above to calculate the required sample size.

*#library(TrialSize)*

*# Define Parameters*

alpha < - 0.025

power < - 0.80

p\_standard < - 0.90 *# Success rate for standard treatment*

p\_new < - 0.90 *# Assumed true success rate for new treatment*

ni\_margin < - 0.10 *# The non-inferiority margin (delta)*

*# Perform Calculation*

ss\_ni\_binary < - TwoSampleProportion.NIS(

p1 = p\_new,

p2 = p\_standard,

delta = p\_new -p\_standard,

alpha = alpha,

beta = 1 - power,

k = 1,

margin=ni\_margin

)

ss\_ni\_binary

## [1] 141

The output shows the sample size needed for each group.

Another R package that may be used to compute sample size for various trial design is “*SampleSize4ClinicalTrials”*. We can use this package for the same example above.

*#library("SampleSize4ClinicalTrials")*

ss\_ni\_binary2 < -ssc\_propcomp(design=3L, *#3L for non-inferiority trial*

ratio=1, alpha=alpha, power=power,

p1 = p\_new,

p2 = p\_standard, delta=ni\_margin)

ss\_ni\_binary2

## Treatment Control

## 1 142 142

The output shows the sample size needed for each group similarly to the above.

#### Time to Event Outcome

Example: a trial comparing a new oral therapy with standard IV chemotherapy for cancer. The new drug is expected to have far fewer side effects, so even if it is slightly less effective, it would be a viable option. The primary outcome is Progression-Free Survival (PFS). The median PFS for the standard IV treatment is estimated to be 11 months. The clinical team has defined the non-inferiority margin (∆) at a Hazard Ratio of 1.30. They are willing to accept the new drug if the hazard of progression is no more than 30% higher than the standard. For the sample size calculation, we will assume the true hazard ratio is 1.0. The trial will recruit patients over 24 months. After the last patient is enrolled, there will be an additional 18 months of follow-up. We desire 90% power with a one-sided alpha of 0.025. The *nSurv()* function of the “*gsDesign”* package [85] may be used.

*#library(gsDesign)*

*# Define Parameters* alpha < - 0.025 power < - 0.90

beta < - 1 - power

*# One-sided significance level*

*# Desired power*

*# Type II error rate*

*# Survival and Non-Inferiority assumptions*

median\_standard < - 11 *# Median PFS in months for standard arm* ni\_margin\_hr < - 1.30 *# Non-inferiority margin (the null HR,* Δ*)*

true\_hr < - 1.0 *# Assumed TRUE hazard ratio*

*# Trial timing assumptions*

accrual\_duration < - 24 *# Months to recruit patients (R)*

follow\_up\_duration < - 18 *# Months of follow-up after last patient in* total\_duration < - accrual\_duration + follow\_up\_duration *# Total duration (T)*

*# Calculate Required Event Rate for the Function -*

*# nSurv requires the exponential event rate (lambda),*

*# not the median survival.*

*# We can calculate this assuming an exponential distribution.*

*# lambda = log(2) / median\_survival*

lambda\_standard < - log(2) / median\_standard

*# Perform the Sample Size Calculation*

ss\_ni\_survival < - nSurv(

lambdaC = lambda\_standard, *# Monthly event rate in standard arm*

hr = true\_hr, *# \*true\* expected HR (usually 1 for NI)*

hr0 = ni\_margin\_hr, *# \*null\* hypothesis HR (NI margin)*

eta = 0, *# Dropout rate (assuming 0 for simplicity)*

R = accrual\_duration, *# Accrual duration*

T = total\_duration, *# Total study duration*

minfup = follow\_up\_duration,

alpha = alpha, *# Already one-sided*

beta = beta, sided = 1

)

ss\_ni\_survival

## Fixed design, two-arm trial with time-to-event

## outcome (Lachin and Foulkes, 1986).

## Solving for: Accrual rate

## Hazard ratio H1/H0=1/1.3

## Study duration: T=42

## Accrual duration: 24

## Min. end-of-study follow-up: minfup=18

## Expected events (total, H1): 613

## Expected sample size (total): 735

## Accrual rates:

## Stratum 1

## 0-24 30.6

## Control event rates (H1):

## Stratum 1

## 0-Inf 0.063

## Censoring rates:

## Stratum 1

## 0-Inf 0

## Power: 100\*(1-beta)=90%

## Type I error (1-sided): 100\*alpha=2.5%

## Equal randomization: ratio=1

The output provides information for planning the trial:

* Total Events (d) is the most important number. The trial needs to continue until

613 events (disease progressions or deaths) have been observed. The final analysis can only happen when this target is met.

* Total Sample Size (n): To accumulate those 613 events within the planned 42-month total study duration, the trial will need to enroll 735 patients in total.
* Accrual Rate (gamma): The trial will need to enroll approximately 30.6 patients per month during 24 months to meet its goals.

## Randomization

The package “*blockrand”* [86] and “*randomizeR”* [87] are among the packages that may be used to generate randomization list for clinical trials. Below are some examples using the “*blockrand”* package for common randomization scenarios.

### Simple Randomization

This is the most straightforward method, akin to flipping a coin for each participant. For example, we do randomization for 100 participants into 2 equal size groups.

*# Set a seed for reproducibility*

set.seed(42)

*# Define trial parameters*

n\_patients < - 100

treatment\_arms < - c("New Drug", "Placebo")

*# Generate the simple randomization list*

randomization\_list\_simple < - data.frame(

patient\_id = 1:n\_patients

) %>%

mutate(

assigned\_arm = sample(x = treatment\_arms,

size = n\_patients,

replace = TRUE,

prob = c(0.5, 0.5))

)

*# View the first 10 assignments*

head(randomization\_list\_simple, 10)

|  |  |  |  |
| --- | --- | --- | --- |
| ## |  | patient\_id | assigned\_arm |
| ## | 1 | 1 | New Drug |
| ## | 2 | 2 | New Drug |
| ## | 3 | 3 | Placebo |
| ## | 4 | 4 | New Drug |
| ## | 5 | 5 | New Drug |
| ## | 6 | 6 | New Drug |
| ## | 7 | 7 | New Drug |
| ## | 8 | 8 | Placebo |
| ## | 9 | 9 | New Drug |
| ## | 10 | 10 | New Drug |

*# Check the final balance*

table(randomization\_list\_simple$assigned\_arm)

##

## New Drug Placebo ## 55 45

The output shows a small imbalance of sample size between arms.

### Blocked Randomization (Permuted Block Randomization)

Blocked randomization is the most common method to prevent the imbalances seen in simple randomization. Using variable block sizes help reduce the risk of predicting the last assignment in a block. The package “*blockrand”* may be used. We first need to define trial parameters, then use variable block sizes to reduce predictability, then input these values to generate the randomization list using the “*blockrand”* package [86].

*# install.packages("blockrand") #library(blockrand)*

*# Set a seed for reproducibility*

set.seed(123)

*# Define trial parameters*

n\_patients < - 120

treatment\_arms < - c("Treatment A", "Control")

*# Use variable block sizes to reduce predictability*

block\_sizes < - c(4, 6, 8)

*# Generate the randomization list using the blockrand package*

randomization\_list\_blocked < - blockrand( n = n\_patients,

num.levels = 2,

levels = treatment\_arms,

block.sizes = c(2, 3, 4), *# \*blockrand\* uses number of subjects per arm in block*

stratum = "Overall" *# use this for stratified randomization later*

)

*# View the first 12 assignments*

head(randomization\_list\_blocked, 12)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | id | stratum | block.id | block.size | treatment |
| ## | 1 1 | Overall | 1 | 8 | Treatment A |
| ## | 2 2 | Overall | 1 | 8 | Treatment A |
| ## | 3 3 | Overall | 1 | 8 | Control |
| ## | 4 4 | Overall | 1 | 8 | Control |
| ## | 5 5 | Overall | 1 | 8 | Control |
| ## | 6 6 | Overall | 1 | 8 | Control |
| ## | 7 7 | Overall | 1 | 8 | Treatment A |
| ## | 8 8 | Overall | 1 | 8 | Treatment A |
| ## | 9 9 | Overall | 2 | 4 | Control |
| ## | 10 10 | Overall | 2 | 4 | Control |
| ## | 11 11 | Overall | 2 | 4 | Treatment A |
| ## | 12 12 | Overall | 2 | 4 | Treatment A |

*# Check the balance after 40 patients*

table(randomization\_list\_blocked$treatment[1:40])

##

## Control Treatment A

## 20 20

*# Check the final balance*

table(randomization\_list\_blocked$treatment)

##

## Control Treatment A

## 60 60

The output shows good balance between two randomized groups.

### Stratified Randomization

Stratified randomization creates a separate block randomization list for each combination of stratification factors, guarantees balance on key prognostic factors, and thus increases statistical power and credibility. It is essential for multi-center trials.

Example: a trial needs to ensure the treatment arms are balanced by Study Site (Site A, Site B) and Disease Stage (Early, Late). The package “*blockrand” [86]* may be used to generate randomization list for each stratum then combine the list for all strata into a master randomization list. In the example codes below, we first define the strata, define trial parameters, use various block sizes, write a loop to generate randomization list for each stratum and combine into the master randomization list, and then check for balance.

*# Load the necessary library #library(blockrand)*

*# Set a seed for reproducibility*

set.seed(101)

*# Define the Strata*

*# Create a data frame where each row is a unique stratum (subgroup)*

strata\_df < - expand.grid(

site = c("Site A", "Site B"),

disease\_stage = c("Early", "Late")

)

print(strata\_df)

## site disease\_stage

## 1 Site A Early

## 2 Site B Early

## 3 Site A Late

## 4 Site B Late

*# Define Trial Parameters*

n\_per\_stratum < - 20 *# We will randomize 20 patients in EACH of the 4 strata*

treatment\_arms < - c("New Drug", "Control")

*# We will use blocks of size 4 and 6 (2 and 3 per arm)*

block\_sizes\_per\_arm < - c(2, 3)

*# Loop Through Strata and Build the Master List*

*# First, create an empty data frame to store the final, combined list*

master\_randomization\_list < - data.frame()

*# Now, loop through each row of the strata\_df (i.e., each stratum)*

for (i in 1:nrow(strata\_df)) {

*# Get the current stratum's information*

current\_site < - strata\_df$site[i]

current\_stage < - strata\_df$disease\_stage[i]

cat(paste("\n--- Generating list for Stratum:",

current\_site, "-", current\_stage, "---\n"))

*# Generate an independent randomization list for ONLY this stratum*

temp\_rand\_list < - blockrand(

n = n\_per\_stratum,

levels = treatment\_arms,

block.sizes = block\_sizes\_per\_arm

)

*# Add columns to "tag" this temporary list with the stratum info* temp\_rand\_list$site < - current\_site

temp\_rand\_list$disease\_stage < - current\_stage

*# Append this temporary list to our master list*

master\_randomization\_list < - rbind(master\_randomization\_list,

temp\_rand\_list)

}

##

## --- Generating list for Stratum: Site A - Early ---

##

## --- Generating list for Stratum: Site B - Early ---

##

## --- Generating list for Stratum: Site A - Late ---

##

## --- Generating list for Stratum: Site B - Late ---

*# View and Verify the Final List*

cat("\n Final Master Randomization List (First 10 rows) \n")

print(head(master\_randomization\_list, 10))

##

## Final Master Randomization List (First 10 rows)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ## |  | id | block.id | block.size | treatment | site | disease\_stage |
| ## | 1 | 1 | 1 | 4 | New Drug | Site A | Early |
| ## | 2 | 2 | 1 | 4 | Control | Site A | Early |
| ## | 3 | 3 | 1 | 4 | Control | Site A | Early |
| ## | 4 | 4 | 1 | 4 | New Drug | Site A | Early |
| ## | 5 | 5 | 2 | 4 | Control | Site A | Early |

cat("\n Final Master Randomization List (Last 10 rows) \n")

print(tail(master\_randomization\_list, 10))

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | 6 | 6 | 2 | 4 | New Drug | Site | A | Early |
| ## | 7 | 7 | 2 | 4 | New Drug | Site | A | Early |
| ## | 8 | 8 | 2 | 4 | Control | Site | A | Early |
| ## | 9 | 9 | 3 | 4 | Control | Site | A | Early |
| ## | 10 | 10 | 3 | 4 | New Drug | Site | A | Early |

##

## Final Master Randomization List (Last 10 rows)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | id | block.id | block.size | treatment | site | disease\_stage |
| ## | 75 11 | 3 | 6 | New Drug | Site B | Late |
| ## | 76 12 | 3 | 6 | Control | Site B | Late |
| ## | 77 13 | 3 | 6 | New Drug | Site B | Late |
| ## | 78 14 | 3 | 6 | Control | Site B | Late |
| ## | 79 15 | 3 | 6 | Control | Site B | Late |
| ## | 80 16 | 3 | 6 | New Drug | Site B | Late |
| ## | 81 17 | 4 | 4 | Control | Site B | Late |
| ## | 82 18 | 4 | 4 | Control | Site B | Late |
| ## | 83 19 | 4 | 4 | New Drug | Site B | Late |
| ## | 84 20 | 4 | 4 | New Drug | Site B | Late |

cat("\n Final Verification of Balance \n") ##

## Final Verification of Balance

*# The final table is the most important check.*

*# It must show perfect balance.*

balance\_table < - table( master\_randomization\_list$treatment, master\_randomization\_list$site, master\_randomization\_list$disease\_stage

)

balance\_table

## , , = Early ##

##

## Site A Site B

## Control 12 10

## New Drug 12 10

##

## , , = Late

##

##

## Site A Site B

## Control 10 10

## New Drug 10 10

The output show good balance between strata and groups.

## Examples of Published Clinical Trial Data Analysis Using R

Below are some journal articles for clinical trials analyzed by the author Nhan Thi Ho et al. using R.

* [Example Clinical Trial Paper 1: “Gatifloxacin versus ceftriaxone for uncomplicated enteric fever in Nepal” published on Lancet Infectious Diseases](https://doi.org/10.1016/s1473-3099(15)00530-7) [89].
* [Example Clinical Trial Paper 2: “A Trial of Itraconazole or Amphotericin B for HIV-Associated Talaromycosis” published on New England Journal of Medicine](https://www.nejm.org/doi/full/10.1056/NEJMoa1613306) [90].

## Case Study: example analysis of a vaccine clinical trial data

The data for the examples below are generated to mimic the data from a multi-centered phase 1/2/3 COVID-19 vaccine trial in which the author Nhan Thi Ho participated as a core researcher and data scientist [91]. This generated dataset are used in the following examples to demonstrate the use of R for clinical trial data analysis and report (although the actual interim and final analyses for this trial for publication and authority approval were done in SAS). The example data contain the following datasets:

* *ran.ex*: randomization list and analysis sets
* *base.ex*: baseline data
* *ve.ex*: time to event outcome (COVID-19) data
* *sae.ex*: serious adverse event data
* *ae.ex*: adverse event data

### Load Required Packages and Example Data

We load the packages "*arsenal*" [19], "*tidyverse*" [11] for baseline, outcome, adverse event summary, "*survival*" [43], "*survminer*" [44] for efficacy analysis of time-to-event outcomes, and load the example data from Github.

*# load multiple packages*

Packages < - c("arsenal", "tidyverse","survival", "survminer")

lapply(Packages, library, character.only = TRUE)

*# Data path*

url\_base < - "https://raw.githubusercontent.com/nhanhocu/"

data\_path < - "biodata-r/main/trialdata.rda"

data\_url < - paste0(url\_base, data\_path)

*# load datasets*

con < - url(data\_url) print(load(con)) close(con)

*# quick look at the datasets*

str(ran.ex)

str(base.ex)

str(ve.ex)

str(sae.ex)

str(ae.ex)

### Baseline Summary

Note that, for clinical trial data, we prepare a variable denoting analysis sets (e.g. intention to treat (ITT), per protocol, safety analysis sets) so that we can easily choose an analysis set for analysis and result display. The first table of a clinical trial paper usually is a summary of baseline characteristics of participants. The examples below are to illustrate the use of *tableby.control()* and *tableby()* of the package “*arsenal*” [19] for descriptive summary tables per treatment arm . The codes are similar to the examples in previous chapters.

*# Set control option for data summary*

d\_controls < - tableby.control(

test = T,

total = F,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = c("meansd", "medianq1q3","meanCI","range","Nmiss2"),

cat.stats = c("countpct", "Nmiss2"),

stats.labels = list(

meansd = "Mean (SD)",

medianq1q3 = "Median (Q1, Q3)", range = "Min - Max",

meanCI = "Mean (95%CI)",

Nmiss2 = "Missing"

)

)

dvar < - c("age","agecat","sex","riskcov")

*# Set labels for variables to be summarized*

mylabels < -as.list(c("Age (year)","Age category (year)",

"Sex","Risk category"))

names(mylabels) < -dvar

*# Create summary table*

tabd < - tableby(as.formula(paste("treatment",paste(dvar,collapse="+"),

sep="~")),

data = subset(base.ex, itt=="1"),

control = d\_controls)

kable(summary(tabd,labelTranslations = mylabels, text=TRUE), caption="Summary of baseline data")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 32**. Summary of baseline data.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Placebo (N = 7345)** | **Vaccine (N = 7753)** | **p Value** |
| Age (year) |  |  | 0.004 |
| - Mean (SD) | 46.539 (13.406) | 45.964 (13.270) |  |
| - Median (Q1, Q3) | 48.000 (37.000, 57.000) | 47.000 (36.000, 56.000) |  |
| - Mean (95%CI) | 46.539 (46.232, 46.846) | 45.964 (45.668, 46.259) |  |
| - Min−Max | 17.000–89.000 | 17.000–85.000 |  |
| - Missing | 0 | 0 |  |
| Age category (year) |  |  | 0.022 |
| - ≤ 60 | 6199 (84.4%) | 6646 (85.7%) |  |
| - >60 | 1146 (15.6%) | 1107 (14.3%) |  |
| - Missing | 0 | 0 |  |
| Sex |  |  | 0.684 |
| - Mean (SD) | 1.645 (11.650) | 1.634 (11.340) |  |
| - Median (Q1, Q3) | 2.000 (1.000, 2.000) | 2.000 (1.000, 2.000) |  |
| - Mean (95%CI) | 1.645 (1.378, 1.911) | 1.634 (1.382, 1.887) |  |
| - Min−Max | 1.000–999.000 | 1.000–999.000 |  |
| - Missing | 0 | 0 |  |
| Risk category |  |  | 0.827 |
| - At\_risk | 3705 (50.4%) | 3897 (50.3%) |  |
| - Healthy | 3640 (49.6%) | 3856 (49.7%) |  |
| - Missing | 0 | 0 |  |

Source: Table by author(s).

### Time-to-Event Outcome Summary

#### Summary with Package “*Arsenal*”

We can do a quick summary of number of event and follow-up time by group for modified intention to treat (mITT) analysis set at Day 92. The function *tableby()* of the package “*arsenal*” [19] may also be used for overall summary and for subgroup stratification (for participants at risk of severe COVID-19).

Overall summary:

o\_controls < - tableby.control(

test = F,

total = F,

numeric.test = "kwt",

cat.test = "chisq",

numeric.stats = "arsenal\_sum",

cat.stats = c("countpct", "Nmiss2"),

stats.labels = list(arsenal\_sum ="Sum"))

ovar < - c("event","ttevent","event3692","ttevent3692")

mylabels < - as.list(c("Event (any time)","Follow-up time (day)",

"Event between Day 36-92", "Follow-up time between 36-92"))

names(mylabels) < -ovar

tabo < - tableby(as.formula(paste("treatment",

paste(ovar,collapse="+"),sep="~")),

data = subset(ve.ex, mitt\_d92=="1"),

control = o\_controls)

kable(summary(tabo,labelTranslations = mylabels, text=TRUE),

caption="Event overall")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 33**. Event overall.

|  |  |  |
| --- | --- | --- |
|  | **Placebo (N = 6825)** | **Vaccine (N = 6849)** |
| Event (any time) |  |  |
| - Sum | 735.000 | 539.000 |
| Follow-up time (day) |  |  |
| - Sum | 618,244.000 | 625,475.000 |
| Event between Day 36–92 |  |  |
| - Sum | 432.000 | 220.000 |
| Follow-up time between 36–92 |  |  |
| - Sum | 601,174.000 | 607,577.000 |

Source: Table by author(s).

Summary stratified by risk category (for participants at risk of severe COVID-19) by defining “*strata=riskcov*” within the *tableby*() function:

tabo < - tableby(as.formula(paste("treatment",

paste(ovar,collapse="+"), sep="~")),

data = subset(ve.ex, mitt\_d92=="1"),

strata=riskcov, *#stratum for stratification*

control = o\_controls)

kable(summary(tabo,labelTranslations = mylabels, text=TRUE),

caption="Event by risk")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 34**. Event by risk.

|  |  |  |  |
| --- | --- | --- | --- |
| **Riskcov** |  | **Placebo (N = 6825)** | **Vaccine (N = 6849)** |
| Healthy | Event (any time) |  |  |
|  | - Sum | 405.000 | 324.000 |
|  | Follow-up time (day) |  |  |
|  | - Sum | 308,028.000 | 309,900.000 |
|  | Event between Day 36–92 |  |  |
|  | - Sum | 222.000 | 133.000 |
|  | Follow-up time between 36-92 |  |  |
|  | - Sum | 297,987.000 | 299,884.000 |
| At\_risk | Event (any time) |  |  |
|  | - Sum | 330.000 | 215.000 |
|  | Follow-up time (day) |  |  |
|  | - Sum | 310,216.000 | 315,575.000 |
|  | Event between Day 36–92 |  |  |
|  | - Sum | 210.000 | 87.000 |
|  | Follow-up time between 36–92 |  |  |
|  | - Sum | 303,187.000 | 307,693.000 |

Source: Table by author(s).

#### Manual Calculation for Later Table Combination

We can do manual summary to make it easier for later combining multiple pieces of summaries into a table.

Summary for all data:

esum < -ve.ex %>% filter(mitt\_d92=="1") %>% group\_by(treatment) %>%

summarise\_at(.vars = vars(event3692, ttevent3692),

.funs = sum) %>%

pivot\_wider(

names\_from = treatment,

values\_from = c(event3692,ttevent3692)

)

esum

## # A tibble: 1 x 4

## event3692\_Placebo event3692\_Vaccine ttevent3692\_Placebo

## <dbl> <dbl> <dbl>

## 1 432 220 601174

## # i 1 more variable: ttevent3692\_Vaccine <dbl>

Summary stratified by risk subgroups (for participants at risk of severe COVID-19) and combining results:

esums < -ve.ex %>%

filter(mitt\_d92=="1") %>% group\_by(treatment,riskcov) %>%

summarise\_at(.vars = vars(event3692, ttevent3692),

.funs = sum) %>%

pivot\_wider(

names\_from = treatment,

values\_from = c(event3692, ttevent3692)

)

esums

## # A tibble: 2 x 5

## riskcov event3692\_Placebo event3692\_Vaccine ttevent3692\_Placebo

## <fct> <dbl> <dbl> <dbl>

## 1 Healthy 222 133 297987

## 2 At\_risk 210 87 303187

## # i 1 more variable: ttevent3692\_Vaccine <dbl>

*#Combined results*

esuma < -bind\_rows(esum, esums)

esuma < -esuma%>%

select(riskcov,contains("Placebo"),contains("Vaccine"))

esuma

## # A tibble: 3 x 5

## riskcov event3692\_Placebo ttevent3692\_Placebo event3692\_Vaccine

## <fct> <dbl> <dbl> <dbl>

## 1 <NA> 432 601174 220

## 2 Healthy 222 297987 133

## 3 At\_risk 210 303187 87

## # i 1 more variable: ttevent3692\_Vaccine <dbl>

### Vaccine Efficacy Cox Model

#### Outcome summary table

In the examples below, we fit Cox models for all data and for each stratum then combine the results (we select analysis set mITT at Day 92).

First, we fit the Cox model and summarize the results for all data:

*# Cox model all*

cfit < -coxph(Surv(ttevent3692,event3692)~treatment,

data=subset(ve.ex, mitt\_d92=="1"))

hrtab < -summary(cfit)$conf.int[,c(1,3:4)]

names(hrtab) < -c("HR","lower.95","upper.95")

*# HR all*

hrtab

## HR lower.95 upper.95

## 0.498 0.424 0.586

*# VE all*

ve < -1-hrtab

ve < -c(ve[1],ve[3],ve[2])

names(ve) < -c("efficacy","lower.95","upper.95")

ve

## efficacy lower.95 upper.95

## 0.502 0.414 0.576

Then we do stratification by risk category (for participants at risk of severe COVID-19) by fitting the Cox model and summarizing the results for each stratum.

At risk participants (*riskcov=At\_risk*):

datr < -subset(ve.ex,mitt\_d92==1 & riskcov=="At\_risk")

fitr < -coxph(Surv(ttevent3692,event3692)~treatment, data=datr)

tabr < -summary(fitr)$conf.int[,c(1,3:4)]

names(tabr) < -c("HR","lower.95","upper.95")

*# HR riskcov=At\_risk*

tabr

## HR lower.95 upper.95

## 0.404 0.314 0.518

*# VE riskcov=At\_risk*

ver < -1-tabr

ver < -c(ver[1],ver[3],ver[2])

names(ver) < -c("efficacy","lower.95","upper.95")

ver

## efficacy lower.95 upper.95

## 0.596 0.482 0.686

Healthy participants (*riskcov=Healthy*):

dath < -subset(ve.ex,mitt\_d92==1 & riskcov=="Healthy")

fith < -coxph(Surv(ttevent3692,event3692)~treatment, data=dath)

tabh < -summary(fith)$conf.int[,c(1,3:4)]

names(tabh) < -c("HR","lower.95","upper.95")

*# HR riskcov=Healthy*

tabh

## HR lower.95 upper.95

## 0.589 0.475 0.730

*# VE riskcov=Healthy*

veh < -1-tabh

veh < -c(veh[1],veh[3],veh[2])

names(veh) < -c("efficacy","lower.95","upper.95")

veh

## efficacy lower.95 upper.95

## 0.411 0.270 0.525

Now, we combine the results from the above outcome summary and from Cox models for all data and each stratum into a nice table:

*#Combine efficacy results*

ve.all < - round(rbind(ve, veh, ver),2)

ve.s < -paste0(ve.all[,1]," (", ve.all[,2],",",ve.all[,3],")")

*#Combine event summary with efficacy results:*

esumas < -cbind(esuma,ve.s)

esumas$riskcov < -c("All participants","- Healthy",

"- At risk of severe COVID-19")

kable(esumas,

caption = "Summary of vaccine efficacy",

col.names = c("Study population",

"Placebo: Total event",

"Placebo: Total follow-up (day)", "Vaccine: Total event",

"Vaccine: Total follow-up (day)", "Vaccine efficacy (95%CI)"))%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 35**. Summary of vaccine efficacy.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Study Population** | **Placebo: Total Event** | **Placebo: Total Follow-Up (Day)** | **Vaccine: Total Event** | **Vaccine: Total Follow-Up (Day)** | **Vaccine Efficacy (95%CI)** |
| All participants | 432 | 601,174 | 220 | 607,577 | 0.5 (0.41,0.58) |
| - Healthy | 222 | 297,987 | 133 | 299,884 | 0.41 (0.27,0.52) |
| - At risk of severe COVID-19 | 210 | 303,187 | 87 | 307,693 | 0.6 (0.48,0.69) |

Source: Table by author(s).

#### Cumulative Incidence Plot

Similar to the examples in previous chapters, we create cumulative incidence plots for treatment groups for all data and stratify by risk groups using the function *surv\_fit()* and *ggsurvplot()* (with *fun = "event"*) of the “*survminer”* package [44].

First, we create *survminer::surv\_fit* model object (note that we clarify *survminer::surv\_fit* to tell R to use specifically the *surv\_fit()* function of the “*survminer”* package) and plot the generated object to create the cumulative incidence plot for all data.

fit < -survminer::surv\_fit(Surv(ttevent3692,event3692)~treatment,

data=subset(ve.ex, mitt\_d92=="1"))

cuminca < -ggsurvplot(fit,

linetype="strata",

conf.int=FALSE,

pval=TRUE,

censor=FALSE,

pval.coord=c(43,0.15),

risk.table = TRUE,

risk.table.height = 0.25,

break.x.by =7, xlim=c(36,92),

fun = "event", legend.title ="",

tables.col = "strata",

legend.labs=c("Placebo","Vaccine"),

xlab="Follow-up day from Day 1",

ylab="COVID-19 cumulative incidence",

risk.table.y.text = FALSE)

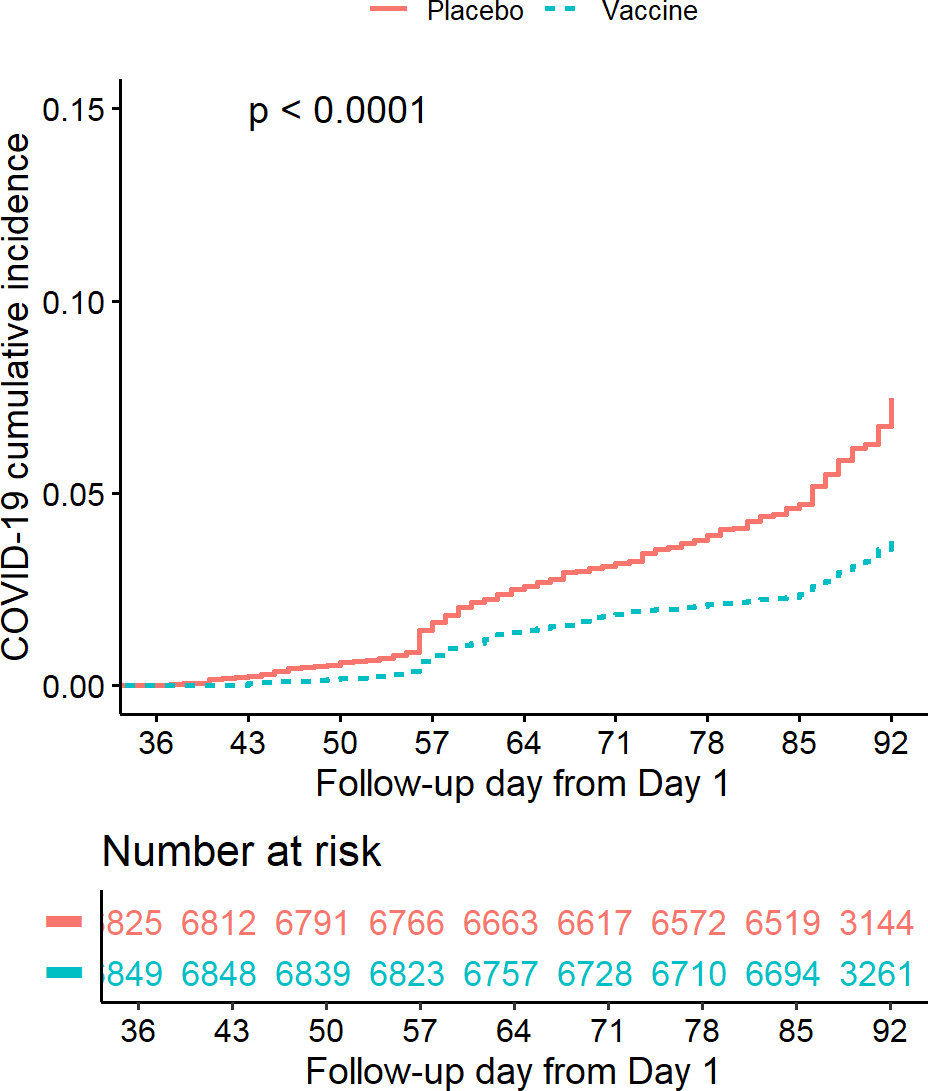
cuminca$plot < -cuminca$plot+

scale\_x\_continuous(breaks = c(0,36,43,50,57,64,71,78,85,92))

cuminca$table < -cuminca$table+

scale\_x\_continuous(breaks = c(0,36,43,50,57,64,71,78,85,92))

cuminca



Source: Figure by author(s).

**Figure 109**. Cumulative incidence plot.

Next, we create the plot stratified by Risk Category by adding “riskcov” to the model and plot the generated *survminer::surv\_fit* object.

fit < -survminer::surv\_fit(Surv(ttevent3692,event3692)~treatment + riskcov, data=subset(ve.ex, mitt\_d92=="1"))

cumincr < -ggsurvplot(fit, linetype="strata", conf.int=FALSE, pval=FALSE, censor=FALSE,

risk.table = TRUE,

risk.table.height = 0.3,

break.x.by =7, xlim=c(36,92), fun = "event", legend.title ="",

legend.labs=c("Placebo.Healthy", "Placebo.At\_risk", "Vaccine.Healthy","Vaccine.At\_risk"),

tables.col = "strata",

xlab="Follow-up day from Day 1",

ylab="COVID-19 cumulative incidence",

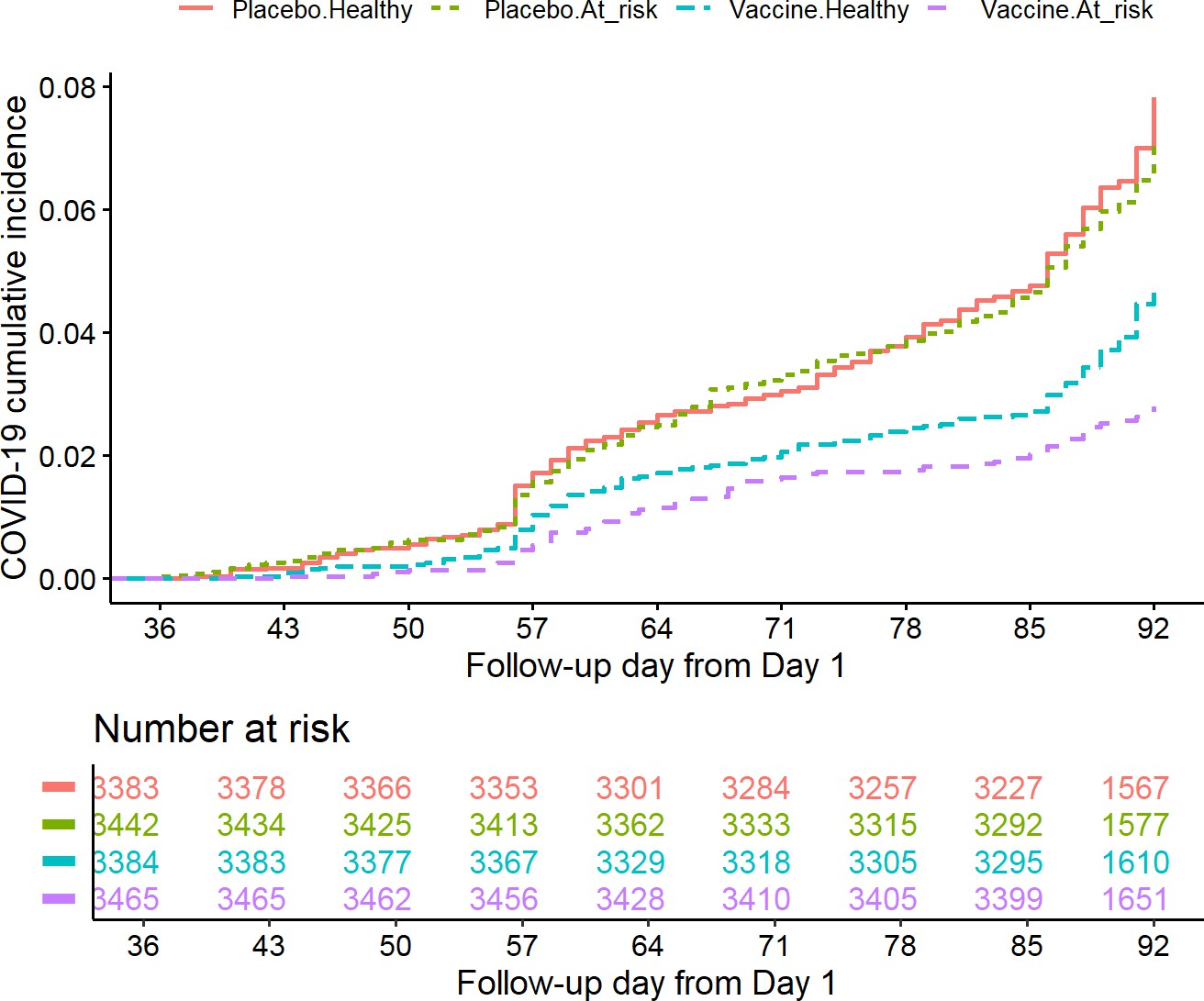
risk.table.y.text = FALSE)

cumincr$plot < -cumincr$plot+

scale\_x\_continuous(breaks = c(0,36,43,50,57,64,71,78,85,92)) cumincr$table < -cumincr$table+

scale\_x\_continuous(breaks = c(0,36,43,50,57,64,71,78,85,92))

cumincr



Source: Figure by author(s).

**Figure 110**. CIF stratified by risk category.

### 10.5.5. Safety Data Summary

Some part of the example R codes below for safety analysis were adapted from the book “[R for Clinical](https://r4csr.org/) [Study Reports and Submission](https://r4csr.org/)” [92]. The “*tidyverse*” functions like *group\_by()*, *summarise()*, *pivot\_longer()* and *pivot\_wider()* are useful for these tasks.

#### Summary of Serious Adverse Events (SAE)

First, we do a summary of Serious Adverse Events (SAE) by Severity, investigational product (IP) Relatedness, and Death on safety analysis set. We calculate the total number of participants in each treatment group:

pop < -ran.ex %>%

filter(sas == "1") %>%

dplyr::count(treatment, name="tot")

pop

## treatment tot ## 1 Vaccine 7753

## 2 Placebo 7345

Then we reformat SAE data and summarize the number and percentage of participants in the safety analysis set who meet each SAE criteria and for each categories: IP relatedness (“drug”), seriousness (“ser”), death (“die”): transform the dataset to long dataset with *pivot\_longer()*, write a simple function to summarize the number and percentage of participants who meet each AE criteria, summarize the number of participants with SAE, then merge with total number of participants to calculate the percentage. The output shows for each category: the number of participants with SAE, the number of participants and percentage:

tidy\_sae < - sae.ex %>% mutate(

all = sas == "1",

drug = aerel %in% "1", ser = aesev == "3",

die = aeout == "1"

) %>%

select(usubjid, treatment, all, drug, ser, die) %>% pivot\_longer(cols = c(all, drug, ser, die))

*#summarize the number and percentage of participants who meet each AE criteria.*

fmt\_num < - function(x, digits, width = digits + 4) { formatC(

x,

digits = digits, format = "f", width = width

)

}

*#summarize sae*

sum.sae < -tidy\_sae %>% filter(value == TRUE) %>% group\_by(treatment, name) %>%

summarise(n = n\_distinct(usubjid))

sum.sae.pop < -merge(sum.sae,pop,by = "treatment", all.x=T) ana < - sum.sae.pop %>%

mutate(

pct = fmt\_num(n / tot \* 100, digits = 1), n = fmt\_num(n, digits = 0),

pct = paste0("(", pct, ")")

)

ana

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | treatment | name | n | tot | pct |
| ## 1 | Placebo | all | 125 | 7345 | ( 1.7) |
| ## 2 | Placebo | die | 6 | 7345 | ( 0.1) |
| ## 3 | Placebo | drug | 7 | 7345 | ( 0.1) |
| ## 4 | Placebo | ser | 31 | 7345 | ( 0.4) |
| ## 5 | Vaccine | all | 130 | 7753 | ( 1.7) |
| ## 6 | Vaccine | die | 9 | 7753 | ( 0.1) |
| ## 7 | Vaccine | drug | 10 | 7753 | ( 0.1) |
| ## 8 | Vaccine | ser | 44 | 7753 | ( 0.6) |

Now we prepare reporting-ready dataset for each AE group.

*# Prepare reporting-ready dataset for each AE group.*

t\_sae < - ana %>% pivot\_wider(

id\_cols = "name",

names\_from = treatment,

values\_from = c(n, pct),

values\_fill = list(

n = " 0",

pct = "( 0.0)"

)

)%>%

mutate(Placebo=paste0(n\_Placebo, " ", pct\_Placebo), Vaccine=paste0(n\_Vaccine, " ", pct\_Vaccine))%>%

select(name,Placebo, Vaccine)

t\_sae

## # A tibble: 4 x 3

## name Placebo Vaccine

## <chr> <chr> <chr>

## 1 all " 125 ( 1.7)" " 130 ( 1.7)"

## 2 die " 6 ( 0.1)" " 9 ( 0.1)"

## 3 drug " 7 ( 0.1)" " 10 ( 0.1)"

## 4 ser " 31 ( 0.4)" " 44 ( 0.6)"

Then we prepare reporting-ready dataset for number of participants by treatment group and combine with the reporting-ready dataset for AE group above:

t\_pop < - pop %>% mutate(name="sas")%>%

pivot\_wider(

names\_from = treatment, values\_from = tot

)%>%

select(name, Placebo, Vaccine)

t\_pop < -lapply(t\_pop,as.character)

dp\_sae < -bind\_rows(t\_pop,t\_sae)%>%

mutate(name = factor(

name,

c("sas","all", "drug", "ser", "die"),

c("Participants in SAS",

"- With one or more SAE", "- With IP-related SAE", "- With grade 3 SAE",

"- With SAE with death outcome"

)

))

dp\_sae

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ## | # | A tibble: | | 5 | x | 3 | | | | | | | | |
| ## |  | name | | |  | |  | Placebo | | | | Vaccine | | |
| ## |  | <fct> | | |  | |  | <chr> | | | | <chr> | | |
| ## | 1 | Participants | | | in | | SAS | "7345" | | | | "7753" | | |
| ## | 2 | - With | one or more SAE | | | | |  | " | 125 | ( 1.7)" | " | 130 | ( 1.7)" |
| ## | 3 | - With | SAE with death | | | | | outcome | " | 6 | ( 0.1)" | " | 9 | ( 0.1)" |
| ## | 4 | - With | IP-related SAE | | | | |  | " | 7 | ( 0.1)" | " | 10 | ( 0.1)" |
| ## | 5 | - With | grade 3 SAE | | | | |  | " | 31 | ( 0.4)" | " | 44 | ( 0.6)" |

We now make a reporting-ready table for the dataset:

kable(dp\_sae,

caption = "Summary of serious adverse events",

col.names = c("Study population",

"Placebo n(%)",

"Vaccine n(%)"))%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 36**. Summary of serious adverse events.

|  |  |  |
| --- | --- | --- |
| **Study Population** | **Placebo n(%)** | **Vaccine n(%)** |
| Participants in SAS | 7345 | 7753 |
| - With one or more SAE | 125 ( 1.7) | 130 ( 1.7) |
| - With SAE with death outcome | 6 ( 0.1) | 9 ( 0.1) |
| - With IP-related SAE | 7 ( 0.1) | 10 ( 0.1) |
| - With grade 3 SAE | 31 ( 0.4) | 44 ( 0.6) |

Source: Table by author(s).

#### Summary of AE by SOC and PT

Safety reporting often includes summarizing Adverse Events (AEs) by System Organ Class (SOC) and Preferred Term (PT) by treatment group. First, we reformat AE data and summarize the number and percentage of participants by SOC: the main logic is to count the unique subjects (*n\_distinct(usubjid)*) who experienced an AE within each treatment and SOC group, and then join this with the total subject count to calculate the percentage.

t1 < - ae.ex %>%

group\_by(treatment, soc) %>%

summarise(n = fmt\_num(n\_distinct(usubjid), digits = 0)) %>%

mutate(pt = soc, order = 0)%>%

mutate(n=as.numeric(n))

t1.pop < -merge(t1,pop,by = "treatment", all.x=T)

t1.np < - t1.pop %>%

mutate(

pct = fmt\_num(n / tot \* 100, digits = 1),

n = fmt\_num(n, digits = 0),

pct = paste0("(", pct, ")")

)

Next, we reformat AE data and summarize the number and percentage of participants by SOC and PT:

t2 < - ae.ex %>%

group\_by(treatment, soc, pt) %>%

summarise(n = fmt\_num(n\_distinct(usubjid), digits = 0)) %>%

mutate(order = 1)%>%

mutate(n=as.numeric(n))

t2.pop < -left\_join(t2, pop, by="treatment")

t2.np < - t2.pop %>%

mutate(

pct = fmt\_num(n / tot \* 100, digits = 1),

n = fmt\_num(n, digits = 0),

pct = paste0("(", pct, ")")

)

head(t2.np)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ##  ## | #  # | A tibble:  Groups: | 6 x 7  treatment, | soc | [2] |  | | | | | | |
| ## |  | treatment | soc |  |  | pt | | n | | order | tot | pct |
| ## |  | <fct> | <chr> |  |  | <chr> | | <chr> | | <dbl> | <int> | <chr> |
| ## | 1 | Vaccine | Blood and lymphatic | | | sy~ | Anae~ | " | ~ | 1 | 7753 | ( 0~ |
| ## | 2 | Vaccine | Blood and lymphatic | | | sy~ | Lymp~ | " | ~ | 1 | 7753 | ( 0~ |
| ## | 3 | Vaccine | Blood and lymphatic | | | sy~ | Lymp~ | " | ~ | 1 | 7753 | ( 0~ |
| ## | 4 | Vaccine | Cardiac disorders | | |  | Acut~ | " | ~ | 1 | 7753 | ( 0~ |
| ## | 5 | Vaccine | Cardiac disorders | | |  | Angi~ | " | ~ | 1 | 7753 | ( 0~ |
| ## | 6 | Vaccine | Cardiac disorders | | |  | Angi~ | " | ~ | 1 | 7753 | ( 0~ |

Then, we combine the above data to prepare reporting data for AE and make a table:

np\_ae < -bind\_rows(t1.np, t2.np) %>%

pivot\_wider(

id\_cols = c(soc, order, pt),

names\_from = treatment,

values\_from = c(n,pct),

values\_fill = fmt\_num(0, digits = 0)

) %>%

arrange(soc, order, pt) %>%

mutate(Vaccine=paste0(n\_Vaccine," ",pct\_Vaccine),

Placebo=paste0(n\_Placebo," ",pct\_Placebo)) %>%

mutate(AE = case\_when(

order==1 ~ paste0("- ",pt),

TRUE ~ pt)) %>%

select(AE,Placebo,Vaccine)

pop.r < -c("Participants in SAS",

pop$tot[pop$treatment %in% "Placebo"],

pop$tot[pop$treatment %in% "Vaccine"])

names(pop.r) < -names(np\_ae)

np\_ae\_pop < -bind\_rows(pop.r, np\_ae)

*# Show first 30 rows*

kable(head(np\_ae\_pop,30), format = "latex", longtabl = TRUE,

col.names = c("Adverse events SOC and PT", "Placebo n (%)", "Vaccine n (%)"),

caption = "Summary of adverse events by SOC and PT")%>%

kable\_styling(latex\_options = c("scale\_down", "hold\_position"))

**Table 37**. Summary of adverse events by SOC and PT.

|  |  |  |
| --- | --- | --- |
| **Adverse Events SOC and PT** | **Placebo n (%)** | **Vaccine n (%)** |
| Participants in SAS | 7345 | 7753 |
| Blood and lymphatic system disorders | 5 ( 0.1) | 6 ( 0.1) |
| - Anaemia | 0 0 | 1 ( 0.0) |
| - Iron deficiency anaemia | 1 ( 0.0) | 0 0 |
| - Lymphadenitis | 2 ( 0.0) | 3 ( 0.0) |
| - Lymphadenopathy | 2 ( 0.0) | 2 ( 0.0) |
| Cardiac disorders | 411 ( 5.6) | 453 ( 5.8) |
| - Acute myocardial infarction | 2 ( 0.0) | 1 ( 0.0) |
| - Angina pectoris | 2 ( 0.0) | 1 ( 0.0) |
| - Angina unstable | 1 ( 0.0) | 1 ( 0.0) |
| - Arrhythmia | 2 ( 0.0) | 1 ( 0.0) |
| - Atrial fibrillation | 1 ( 0.0) | 0 0 |
| - Atrioventricular block complete | 0 0 | 1 ( 0.0) |
| - Bradycardia | 4 ( 0.1) | 6 ( 0.1) |
| - Chest discomfort | 11 ( 0.1) | 11 ( 0.1) |
| - Chest pain | 8 ( 0.1) | 14 ( 0.2) |
| - Dizziness | 19 ( 0.3) | 28 ( 0.4) |
| - Dyspnoea | 9 ( 0.1) | 4 ( 0.1) |
| - Heart valve incompetence | 0 0 | 1 ( 0.0) |
| - Mitral valve incompetence | 1 ( 0.0) | 0 0 |
| - Myocardial ischaemia | 3 ( 0.0) | 5 ( 0.1) |
| - Palpitations | 0 0 | 1 ( 0.0) |
| - Sinus arrhythmia | 0 0 | 1 ( 0.0) |
| - Syncope | 2 ( 0.0) | 1 ( 0.0) |
| - Tachycardia | 352 ( 4.8) | 382 ( 4.9) |
| Congenital, familial and genetic disorders | 1 ( 0.0) | 0 0 |
| - Atrial septal defect | 1 ( 0.0) | 0 0 |
| Ear and labyrinth disorders | 25 ( 0.3) | 43 ( 0.6) |
| - Ear infection | 1 ( 0.0) | 2 ( 0.0) |
| - Ear pain | 3 ( 0.0) | 4 ( 0.1) |

Source: Table by author(s).

Above are some example codes for analysis and report of example clinical trial data. Ones may apply what have been learnt from previous chapters for further analysis and visualization.

## Useful Resources

### Books, Courses

* + - * [Fundamentals of Clinical Trials](https://www.springer.com/gp/book/9783319185385) [93].
      * [Sample Size Calculation in Clinical Research](https://www.taylorfrancis.com/books/mono/10.1201/9781315183084/sample-size-calculations-clinical-research-shein-chung-chow-jun-shao-hansheng-wang-yuliya-lokhnygina) [88].
      * [Courses: Design and Analysis of Clinical Trials](https://online.stat.psu.edu/stat509/node/164/) (https://online.stat.psu.edu/stat509/node/164/).
      * [R for Clinical Study Reports and Submission](https://r4csr.org/) [92].

### Journal Articles

* + - * [Example Clinical Trial Paper 1 published on Lancet Infectious Diseases](http://dx.doi.org/10.1016/S1473-3099(15)00530-7) [89].
      * [Example Clinical Trial Paper 2 published on New England Journal of Medicine](https://www.nejm.org/doi/full/10.1056/NEJMoa1613306) [90].
      * [Example Clinical Trial Paper 3 published on Lancet Global Health](https://doi.org/10.1016/S2214-109X(19)30422-X) [94].
      * [Example Clinical Trial Paper 4 published on Nature Communications](https://doi.org/10.1038/s41467-024-47905-1) [91].

## Chapter Summary

* *“pwr”, “powerSurvEpi”, “TrialSize”, “SampleSize4ClinicalTrials”, “gsDesign”* are main packages for sample size - power estimation for various clinical trial designs and outcome types
* *“blockrand”* and “*randomizeR”* packages may be used for various randomization approaches
* Main analysis and report for clinical trial data:
  + Baseline summary (e.g., package “*arsenal”*)
  + Outcome evaluation (various packages, approaches from previous chapters: e.g., “*survival”, “survminer”* for time to event outcome),
  + Adverse event summary (e.g., package collection “*tidyverse”*)

# Analysis of Microbiome Data with “*metamicrobiomeR”*

The “*metamicrobiomeR”*package was developed by Nhan Thi Ho, the author of this book. During her research project at Columbia University Medical Center regarding meta-analysis of effects of exclusive breastfeeding on infant gut microbiome across populations, many difficulties arose. First, there was a lack of statistical method for proper analysis of zero-inflated compositional microbiome relative abundance data. Second, available methods for differential analysis of microbiome data did not produce standardized effect estimates that might be pooled across studies for meta-analysis. To address these gaps, she implemented Generalized Additive Model for Location, Scale and Shape (GAMLSS) with zero inflated beta (BEZI) family for analysis of microbiome relative abundance data with various options for data transformation or normalization to address compositional effects. Moreover, the differential effect estimates (log(odds ratio)) using this method may be pooled across studies for meta-analysis. The “*metamicrobiomeR”*package is the implementation of these approaches together with some other related functions for microbiome data analysis, meta-analysis and visualization. The paper describing the “*metamicrobiomeR”*package may be found on [BMC Bioinformatics](https://doi.org/10.1186/s12859-019-2744-2) [95]. The package is hosted in [R cran](https://cran.r-project.org/web/packages/metamicrobiomeR/index.html) [4] and an earlier version with full package tutorial may be found on [Github](https://github.com/nhanhocu/metamicrobiomeR).

This chapter will guide through some main steps for microbiome data analysis and meta-analysis using the “*metamicrobiomeR”* package.

## Statistical Approaches

The R package “*metamicrobiomeR”*applies Generalized Additive Models for Location, Scale and Shape (GAMLSS) with a zero-inflated beta (BEZI) family (GAMLSS-BEZI) for the analysis of microbial taxonomy relative abundance data. GAMLSS is a general framework for fitting regression type models in which the response variable can be any distribution [96]. With BEZI family, this model allows direct and proper examination of microbiome relative abundance data, which resemble a zero-inflated beta distribution. In principle, this model is similar to the two-part mixed effect model proposed by Chen et al.[97] in that the presence or absence of the taxon in the samples is modeled with a logistic component and the non-zero abundance of the taxon is modeled with a Beta component. Both logistic and beta components allow covariate adjustment and address longitudinal correlations with subject-specific random effects.

The GAMLSS-BEZI is based on the broadly applicable established GAMLSS framework that can be flexibly implemented and applied to different types of data and study designs (e.g., cross-sectional and longitudinal) as in the *taxa.compare()* functions. Importantly, the estimates (regression coefficients) from GAMLSS-BEZI are log (odds ratio) of being in the case group (as compared to being in the control group) with changes in relative abundance of a specific bacterial taxon and thus are analogous across microbiome studies and can be directly combined using standard meta-analysis approaches. As such, random effects or fixed effects meta-analysis models may be applied to pool the estimates and standard errors as implemented in *meta.taxa()* functions of the “*metamicrobiomeR”*package. This approach allows examination of study-specific effects, heterogeneity between studies, and the overall pooled effects across studies. In the *taxa.compare()* function, all bacterial taxa or pathway data are first filtered to retain features with mean relative abundance

≥ relative abundance threshold (e.g., ≥0.005%) and with prevalence ≥ prevalence threshold (e.g., present in ≥5% of the total number of samples). This pre-filtering step has been shown to improve performance of various differential abundance

detection strategies [98]. A filtered data matrix is then modeled by GAMLSS-BEZI and (*µ*) logit link and other default options using the R package “*gamlss*” version 5.0–5 [74].

For longitudinal data, subject-specific random effects can be added to the model. The default implementation is subject random intercepts as in practice this is often sufficient to address the longitudinal correlations [99]. However, it is possible to extend the model to include random slopes depending on the specific research content.

In addition, different approaches to deal with compositional effects are also implemented including Centered Log Ratio (CLR) transformation [100] with various zero-replacement options [101] and Geometric Mean of Pairwise Ratios (GMPR) normalization [102]. Multiple testing adjustment can be done using different methods (False Discovery Rate (FDR) control by default). For more detailed description of statistical approaches and demonstration, please see the package paper at [BMC](https://doi.org/10.1186/s12859-019-2744-2) [Bioinformatics](https://doi.org/10.1186/s12859-019-2744-2) [95].

## Example Data

For differential relative abundances analysis, the example dataset is published data from a cohort study of 50 healthy Bangladeshi infants, which included longitudinal gut microbiome data from 996 stool samples collected monthly from birth to 2 years of life [13]. This study included longitudinal monthly data regarding the infants’ breastfeeding practices (exclusive, non-exclusive), duration of exclusive breastfeeding, infant age (months) at solid food introduction, and occurrence of diarrhea around the time of stool sample collection. A subset of samples collected around birth are used as example cross-sectional dataset (50 samples) and data from all samples are used as a longitudinal dataset (996 samples). For meta-analysis, gut microbiome data from four published studies are used to demonstrate the application of random effects models for meta-analysis across microbiome studies. These four studies include:

* A cohort of healthy infants in Bangladesh [13] (the data of this study was also used in the examples demonstrating the use of GAMLSS-BEZI with *taxa.compare()* function);
* A cross-sectional study of Haiti infants negative for HIV who were exposed or unexposed to maternal HIV [103];
* A cohort of healthy infants in the USA (California and Florida [CA\_FL]) [104]; and
* A small cohort of healthy infants in the USA (North Carolina [NC]) [105].

More details about the four studies included in the meta-analysis are described in the “*metamicrobiomeR*” [package paper](https://doi.org/10.1186/s12859-019-2744-2) [95].

## *“metamicrobiomeR”* Package Summary

The “*metamicrobiomeR”*package includes the functions in the table below.

**Table 38**. Functions in the “metamicrobiomeR” package.

**Functions Description**

*taxa.filter* Filter relative abundances of bacterial taxa or pathways using prevalence and abundance thresholds

*taxa.meansdn* Summarize mean, standard deviation of abundances and number of subjects by groups for all bacterial taxa or pathways

*taxa.mean.plot* Plot mean abundance by groups (from taxa.meansdn output) *taxa.compare* Compare relative abundances of bacterial taxa at all levels using GAMLSS

or LM or LMAS

*pathway.compare* Compare relative abundances of bacterial functional pathways at all levels using GAMLSS or LM or LMAS. Compare log(absolute abundances) of bacterial functional pathways using LM

*taxcomtab.show* Display the results of relative abundance comparison (from taxa.compare or pathway.compare outputs)

*meta.taxa* Perform meta-analysis of relative abundance estimates of bacterial taxa or pathways across studies using random/fixed effects

*metatab.show* Display meta-analysis results of bacterial taxa or pathway relative abundances (from meta.taxa output)

*meta.niceplot* Produce combined heatmap and forest plot for meta-analysis results of bacterial taxa and pathway relative abundances

*read.multi* Read multiple files in a path to R

*alpha.compare* Calculate average alpha diversity indexes for a specific rarefaction depth, standardize and compare indexes between groups

*microbiomeage* Predict microbiome age using Random Forest model based on bacterial genera abundances

Source: Table by author(s).

## Package Usage

This is a short, simplified version of the package tutorial. Only illustration and examples for the analysis of differential relative abundance using GAMLSS-BEZI and meta-analysis across studies using random effects models are shown. Full illustration, examples of all implemented functions, workflows and data are available at the package [Github repo](https://github.com/nhanhocu/metamicrobiomeR)sitory. A simplified version of package vignette is also available in [R cran](https://cran.r-project.org/web/packages/metamicrobiomeR/vignettes/my-vignette.html) [4].

First, we install and load the “*metamicrobiomeR”*package and other required packages (e.g. “*gdata*” [106]):

*#install.packages(“metamicrobiomeR”)*

library(metamicrobiomeR)

*# Load other required packages*

library(gdata)

### Example Differential Analysis: Comparison Between Breastfeeding Statuses

The following example demonstrates comparing the relative abundance of bacterial taxa between infants (< 6 Months of Age) who were exclusively breastfed versus those who were non-exclusively breastfed (the differential analysis).

In the example data, relative abundances of bacterial taxa at various taxonomic levels (from phylum to genus or species) are obtained via the *summarize\_taxa.py* script in QIIME1 [107]. Bacterial functional pathway abundances (e.g., Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway level 1 to 3) are obtained from metagenome prediction analysis using PICRUSt [108]. The relative abundance data should first be filtered with *taxa.filter()* function. Then the function *taxa.meansdn()* is used to summarize the mean, standard deviation of abundances and number of subjects by groups for all bacterial taxa of all levels or for pathways by comparison and grouping variables. Then the function *taxa.mean.plot()* is used to visualize data summary for selected taxonomic level as in the example codes and figure. The values to be displayed in the figure may be filtered by *mean.filter* option and taxonomic level names may be displayed as short or full long names via option *show.taxname*.

#### Plot relative abundance

Now, we create a plot of Mean Relative Abundance by Breastfeeding Statuses and AGE at phylum level (*tax.lev="l2"*):

*# Get and take a quick look at the data provided by*

*# the "metamicrobiomeR" package*

data(taxtab6)

*#str(taxtab6)*

*# data summary and plot*

taxlist.rm < -taxa.filter(taxtab=taxtab6,

percent.filter = 0.05,

relabund.filter = 0.00005) taxa.meansdn.rm < -taxa.meansdn(taxtab=taxtab6,

sumvar="bf", groupvar="age.sample")

taxa.meansdn.rm < -taxa.meansdn.rm[taxa.meansdn.rm$bf!="No\_BF",]

taxa.meansdn.rm$bf < -gdata::drop.levels(taxa.meansdn.rm$bf,

reorder=FALSE)

*#phylum*

p.bf.l2 < -taxa.mean.plot(tabmean=taxa.meansdn.rm,

tax.lev="l2", comvar="bf", groupvar="age.sample", mean.filter=0.005, show.taxname="short")

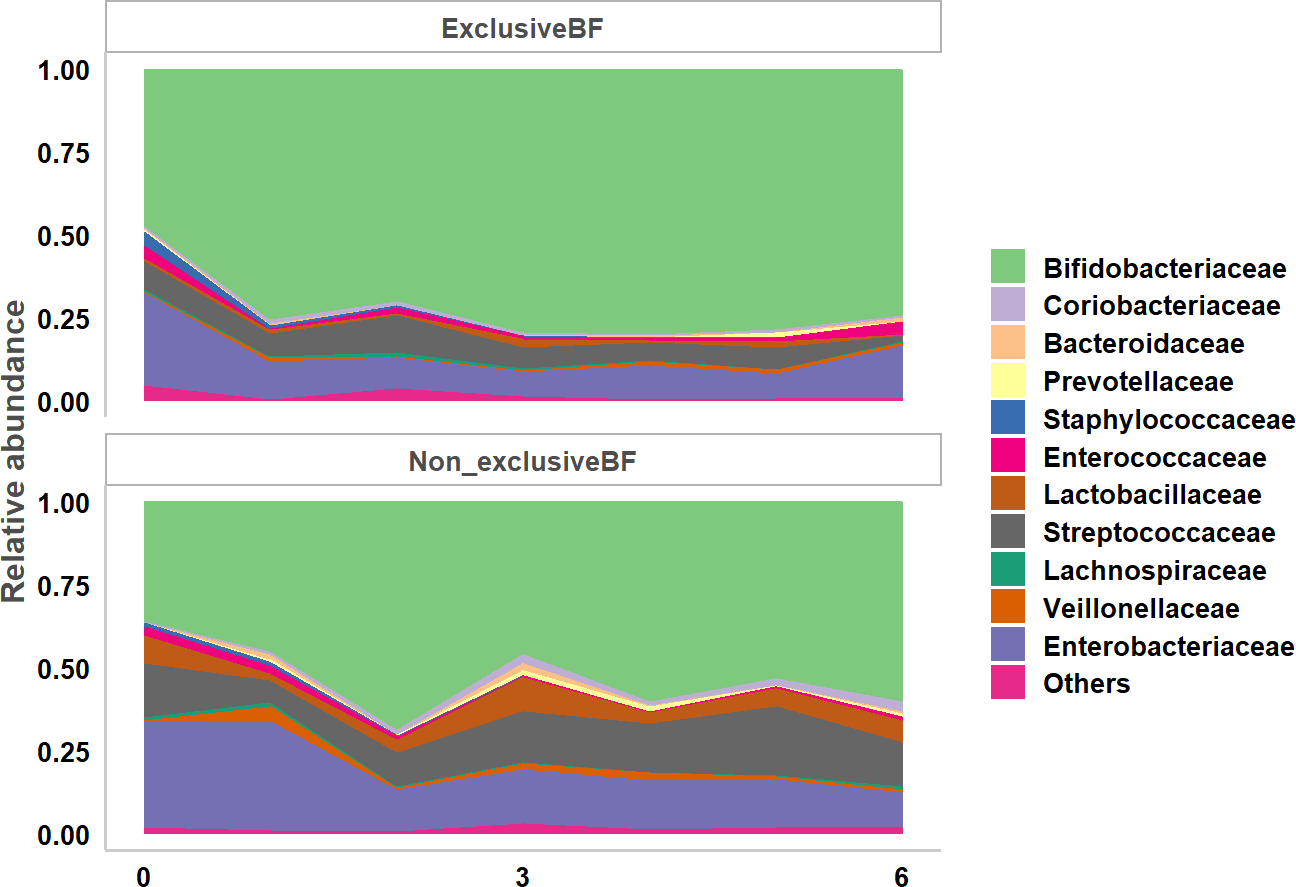
p.bf.l2$p



Source: Figure by author(s).

**Figure 111**. Taxa mean plot phylum.

To create a plot at family level, we change to *tax.lev="l5"*:



p.bf.l5 < -taxa.mean.plot(tabmean=taxa.meansdn.rm,

tax.lev="l5", comvar="bf", groupvar="age.sample", mean.filter=0.005, show.taxname="short")

p.bf.l5$p

Source: Figure by author(s).

**Figure 112**. Taxa mean plot family.

#### Comparison of Taxonomic Data between Groups Using GAMLSS-BEZI

*taxa.compare()* and *pathway.compare()* are basically similar. There are just some adjustments to make these two functions readily to be used with different data format for pathway and taxonomy data. We need to specify the taxonomy dataset to be analyzed with *taxtab*, comparison variable with *comvar* and variables to be adjusted with *adjustvar*. Various statistical methods for differential analysis of relative abundance data may be selected by *propmed.rel* with default= “*gamlss*”. Various data transformation may be selected by *transform* with default= “*none*”. Various methods for zero replacement may be selected by *zeroreplace.method* with default = “*none*”. The data may be filtered before differential analysis by setting filter threshold for prevalence with *percent.filter* and for relative abundance with *relabund.filter*. If the data is longitudinal, select the option *longitudinal* = “Yes”. Various options for adjusting for multiple testing may be selected via option *p.adjust.method* with default = “*fdr*”.

Below is an example for comparison between breastfeeding statuses adjusting for age of infants at sample collection. Of note, the running time is not long in regular laptop for both GAMLSS-BEZI analysis (~10s) and meta-analysis (~5s). However, to save running time, only taxonomies of one phylum are selected for differential analysis example.

tab6 < -as.data.frame(taxtab6)

tl < -colnames(taxtab6)[grep("k bacteria.p fusobacteria",

colnames(taxtab6))]

taxacom.ex < -taxa.compare(taxtab =

tab6[,c("personid","x.sampleid", "bf","age.sample",

tl)], propmed.rel="gamlss",

comvar="bf",

adjustvar="age.sample",

longitudinal="yes",

p.adjust.method="fdr")

The result output from the *taxa.compare()* function may be displayed using *taxcomtab.show()* function. The taxa level to be displayed may be selected by *tax.lev* with options from “*l2*” (phylum) to “*l7*” (species) and the default is “*l2*”. Options for multiple testing re-adjustment for only the level to be displayed or keeping original multiple testing adjustment for all taxa of all levels may be selected with *readjust.p*. In the example below, we display the results for the comparison between non-exclusive vs. exclusive breastfeeding (BF) (*showvar="bfNon\_exclusiveBF"*) at phylum level (*tax.lev="l2"*).

tp < -taxcomtab.show(taxcomtab=taxacom.ex,

tax.select="none", showvar="bfNon\_exclusiveBF",

tax.lev="l2",

readjust.p=TRUE,

p.adjust.method="fdr",

p.cutoff = 1)

tp

## id Estimate.bfNon\_exclusiveBF ll

## 1 k bacteria.p fusobacteria -0.06 -0.77

## ul Pr(>|t|).bfNon\_exclusiveBF pval.adjust.bfNon\_exclusiveBF

## 1 0.65 0.87 0.87

The output shows the estimate (estimated Log Odds Ratio: “*Estimate.bfNon\_exclusiveBF*”), 95%CI lower limit (“*ll*”), upper limit (“*ul*”), p-value (“*Pr(>|t|).bfNon\_exclusiveBF*”) and multiple testing adjusted p-value (“*pval.adjust.bfNon\_exclusiveBF*”).

We can display the results of genus level by specifying *tax.lev="l6"*:

tp < -taxcomtab.show(taxcomtab=taxacom.ex,

tax.select="none", showvar="bfNon\_exclusiveBF",

tax.lev="l6",

readjust.p=TRUE,

p.adjust.method="fdr",

p.cutoff = 1)

head(tp)

##

## 5 k bacteria.p fusobacteria.c fusobacteriia.o fusobacteriales. f fusobacteriaceae.g fusobacterium

## Estimate.bfNon\_exclusiveBF ll ul ## 5 -0.12 -0.95 0.72

## Pr(>|t|).bfNon\_exclusiveBF pval.adjust.bfNon\_exclusiveBF

## 5 0.785 0.785

#### Comparison Taxonomic Data Between Groups Using Linear Model with Arcsin Squareroot Transformation (LMAS)

This approach was formerly used with MaAsLin by [The Huttenhower Lab](https://huttenhower.sph.harvard.edu/maaslin/) [109]. This is also implemented in the “*metamicrobiomeR”*package for comparison and demonstration of the advantage of the GAMLSS-BEZI approaches for microbiome relative abundance data. In the below example, we do comparison of taxonomic data between breastfeeding statuses adjusting for age of infants at sample collection and display the results of phylum and family levels:

*#phylum*

ts < -taxcomtab.show(taxcomtab=taxacom.lmas,

tax.select="none", showvar="bfNon\_exclusiveBF",

tax.lev="l2",

readjust.p=TRUE,

p.adjust.method="fdr",

p.cutoff = 1,

digit=5, p.digit=5)

ts

taxacom.lmas < -taxa.compare(taxtab =

tab6[,c("personid","x.sampleid","bf","age.sample", tl)],

propmed.rel="lm", transform="asin.sqrt", comvar="bf", adjustvar="age.sample", longitudinal="yes", p.adjust.method="fdr")

## id Estimate.bfNon\_exclusiveBF

## 1 k bacteria.p fusobacteria 0.00134

## ll ul Pr(>|t|).bfNon\_exclusiveBF

## 1 -0.00134 0.00402 0.321

## pval.adjust.bfNon\_exclusiveBF

## 1 0.321

*#family*

ts < -taxcomtab.show(taxcomtab=taxacom.lmas,

tax.select="none", showvar="bfNon\_exclusiveBF", tax.lev="l5",

readjust.p=TRUE, p.adjust.method="fdr",

p.cutoff = 1,

digit=5, p.digit=5)

head(ts)

##

## 4 k bacteria.p fusobacteria.c fusobacteriia.o fusobacteriales. f fusobacteriaceae

## Estimate.bfNon\_exclusiveBF ll ul ## 4 0.00137 -0.00131 0.00405

## Pr(>|t|).bfNon\_exclusiveBF pval.adjust.bfNon\_exclusiveBF

## 4 0.31 0.31

#### Comparison of Bacterial Functional (KEGG) Pathways Between Groups

The approach in *pathway.compare()* is similar to that in *taxa.compare()* function except some adjustment for data format obtained from metagenome prediction analysis using PICRUSt [108].

In the below example, we do comparison of relative abundance of bacterial functional (KEGG) pathways between male vs. female infants < 6 months adjusting for breastfeeding statuses and age of infants at sample collection. We will do the analysis and display the results for pathway relative abundances for level 1 only (to save running time). We need to specify the pathway relative abundance data (*pathtab=kegg.12*), the covariate data (*mapfile=covar.rm*), the id variable to merge the two data files (*sampleid="sampleid"*), that we want to do the comparison for relative abundance (*pathsum="rel"*), and other options similar to those of the *taxa.compare()* function above.

*# Get the data and take a quick look at the data*

data(kegg.12)

*#str(kegg.12)*

data(covar.rm)

*#str(covar.rm)*

*# Compare between groups*

path1 < -pathway.compare(pathtab=kegg.12,

mapfile=covar.rm,

sampleid="sampleid",

pathsum="rel",

stat.med="gamlss",

comvar="gender", adjustvar=c("age.sample","bf"), longitudinal="yes",

p.adjust.method="fdr",

percent.filter=0.05,

relabund.filter=0.00005)

We then use the same function *taxcomtab.show()* to display the results of KEGG pathway level 1 and level 2. Note that we specify *tax.lev="l2"* and change the dataset option *taxcomtab=path1$l1* or *taxcomtab=path1$l2* to display the results of level 1 or level 2 pathway.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *#Level 1*  tk < -taxcomtab.show(taxcomtab=path1$l1, | | | | |  | |
|  |  | sumvar="path", | | |
|  |  | tax.lev="l2", | | |
|  |  | tax.select="none", | | |
|  |  | showvar="genderMale", | | |
|  |  | p.adjust.method="fdr", | | |
|  |  | p.cutoff=1) | | |
| tk |  |  | | |
| ## |  | id | | | Estimate.genderMale | ll |
| ## | 7 | Organismal.Systems | | | 0.04 | 0.02 |
| ## | 5 | Metabolism | | | 0.01 | 0.00 |
| ## | 4 | Human.Diseases | | | -0.02 | -0.03 |
| ## | 8 | Unclassified | | | -0.01 | -0.02 |
| ## | 2 | Environmental.Information.Processing | | | -0.01 | -0.02 |
| ## | 3 | Genetic.Information.Processing | | | 0.01 | 0.00 |
| ## | 6 | None | | | -0.06 | -0.16 |
| ## | 1 | Cellular.Processes | | | 0.00 | -0.03 |
| ## |  | ul | Pr(>|t|).genderMale | pval.adjust.genderMale | | |
| ## | 7 | 0.06 | 0.0000 | 0.0001 | | |
| ## | 5 | 0.01 | 0.0155 | 0.0619 | | |
| ## | 4 | 0.00 | 0.0273 | 0.0727 | | |
| ## | 8 | 0.00 | 0.0521 | 0.1043 | | |
| ## | 2 | 0.00 | 0.1705 | 0.2728 | | |
| ## | 3 | 0.01 | 0.2454 | 0.3271 | | |
| ## | 6 | 0.05 | 0.2877 | 0.3288 | | |
| ## | 1 | 0.02 | 0.8100 | 0.8100 | | |

*#Level 2*

tk < -taxcomtab.show(taxcomtab=path1$l2,

sumvar="path",

tax.lev="l2",

tax.select="none", showvar="genderMale", p.adjust.method="fdr",

p.cutoff=1)

head(tk)

|  |  |  |
| --- | --- | --- |
| ## |  | id |
| ## | 36 | Cellular.Processes..Transport.and.Catabolism |
| ## | 9 | Organismal.Systems..Endocrine.System |
| ## | 2 | Metabolism..Biosynthesis.of.Other.Secondary.Metabolites |
| ## | 34 | Genetic.Information.Processing..Transcription |
| ## | 1 | Metabolism..Amino.Acid.Metabolism |
| ## | 20 | Metabolism..Lipid.Metabolism |
| ## |  | Estimate.genderMale ll ul Pr(>|t|).genderMale |
| ## | 36 | 0.07 0.04 0.10 0.0000 |
| ## | 9 | 0.09 0.05 0.12 0.0000 |
| ## | 2 | 0.03 0.02 0.04 0.0000 |
| ## | 34 | -0.02 -0.03 -0.01 0.0000 |
| ## | 1 | 0.02 0.01 0.03 0.0001 |
| ## | 20 | -0.01 -0.02 0.00 0.0004 |
| ## |  | pval.adjust.genderMale |
| ## | 36 | 0.0000 |
| ## | 9 | 0.0000 |
| ## | 2 | 0.0001 |
| ## | 34 | 0.0001 |
| ## | 1 | 0.0006 |
| ## | 20 | 0.0026 |

### Illustration of Meta-Analysis

#### Metaanalysis of Taxonomy or Pathway Relative Abundance Data

Meta-analysis is illustrated with the example comparing relative abundances of gut bacterial taxa and bacterial predicted functional pathways between male vs. female infants ≤6 months of age adjusting for feeding status and infant age at the time of stool sample collection. GAMLSS-BEZI results from four studies (Bangladesh, Haiti, USA(CA\_FL), USA(NC)) were used for meta-analysis with the *meta.taxa()* function. Summary measures may be selected by *summary.measure* with default= “RR” for estimates from GAMLSS with BEZI family and should be “RD” for estimates from Linear/linear mixed effect model. The threshold percentage of number of studies that a taxon is available to do metaanalysis may be selected by *percent.meta*. The result output from *meta.taxa()* function may be displayed as a table using the *metatab.show()* function and then visualized as combined heatmap and forest plot using the *meta.niceplot()* function. The example plot shows heatmap of log(odds ratio) (log(OR)) of relative abundances of gut bacterial taxa at different taxonomic levels between male vs. female infants for each study and pooled estimates (meta-analysis) across all studies with 95% confidence intervals (95% CI) (forest plot). All log(OR) estimates of each bacterial taxa from each study were from Generalized Additive Models for Location Scale and Shape (GAMLSS) with beta zero inflated family (BEZI) and were adjusted for feeding status and age of infants at sample collection. Pooled log(OR) estimates and 95% CI (forest plot) were from random effect meta-analysis models with inverse variance weighting and DerSimonian-Laird estimator for between-study variance based on the adjusted log(OR) estimates and corresponding standard errors of all included studies. Bacterial taxa with *p*-values for differential relative abundances < 0.05 were denoted with \* and those with *p*-values < 0.0001 were denoted with \*\*. Pooled log(OR) estimates with pooled *p*-values < 0.05 are in red and those with false discovery rate (FDR) adjusted pooled *p*-values < 0.1 are in triangle shape. Missing (unavailable) values are in white. (Abbreviation: USA: United States of America; CA: California; FL: Florida; NC: North Carolina).

First, we load saved GAMLSS-BEZI results of four studies for the comparison of bacterial taxa relative abundance between genders adjusted for breastfeeding and infant age at sample collection and perform meta-analysis. We select only taxonomies of a phylum for meta-analysis example (to save running time) and show meta-analysis results in a table. We show the example results for phylum (*tax.lev="l2"*) and family level(*tax.lev="l5"*).

*# Get and take a quick look at the dataset*

data(tabsex4)

str(tabsex4)

## 'data.frame': 701 obs. of 23 variables:

## $ id : chr "k bacteria.p actinobacteria" "k bacteria. p bacteroidetes" "k bacteria.p firmicutes"

"k bacteria.p fusobacteria" ...

## $ Estimate.genderMale : num 0.2269 -0.1549 -0.056 -0.0648 -0.0887 ...

## $ Std. Error.genderMale : num 0.121 0.119 0.104 0.282 0.118 ...

## $ t value.genderMale : num 1.874 -1.305 -0.537 -0.23 -0.751 ...

## $ Pr(>|t|).genderMale : num 0.0619 0.1931 0.5919 0.8186 0.4532 ...

## $ Estimate.bfNon\_exclusiveBF : num -0.3688 0.2557 0.2362 -0.0605 0.3699 ...

## $ Std. Error.bfNon\_exclusiveBF : num 0.139 0.136 0.116 0.364 0.134 ...

## $ t value.bfNon\_exclusiveBF : num -2.647 1.887 2.035 -0.166 2.764 ...

## $ Pr(>|t|).bfNon\_exclusiveBF : num 0.00857 0.0602 0.04276 0.86812 0.00606 ...

## $ Estimate.bfNo\_BF : num -1.329 0.135 0.356 -0.403 1.153 ...

## $ Std. Error.bfNo\_BF : num 0.412 0.499 0.392 1.083 0.407 ...

## $ t value.bfNo\_BF : num -3.223 0.27 0.908 -0.372 2.834 ...

## $ Pr(>|t|).bfNo\_BF : num 0.00141 0.7873 0.36478 0.7103 0.0049 ...

## $ Estimate.age.sample : num 0.1251 0.0242 0.0115 0.0285 -0.1348 ...

## $ Std. Error.age.sample : num 0.0377 0.0367 0.0319 0.0857 0.038 ...

## $ t value.age.sample : num 3.318 0.66 0.359 0.333 -3.549 ...

## $ Pr(>|t|).age.sample : num 0.001022 0.510068 0.719525 0.73934

0.000448 ...

## $ pval.adjust.genderMale : num 0.48 0.899 0.961 0.961 0.961 ...

## $ pval.adjust.bfNon\_exclusiveBF: num 0.059 0.2333 0.1767 0.9976 0.0448 ...

## $ pval.adjust.bfNo\_BF : num 0.0219 1 1 1 0.0468 ...

## $ pval.adjust.sample : num 0.00786 0.82595 0.85375 0.85375 0.00505 ... ## $ study : chr "Subramanian et al 2014 (Bangladesh)"

"Subramanian et al 2014 (Bangladesh)" "Subramanian et al 2014 (Bangladesh)" "Subramanian et al 2014 (Bangladesh)" ...

## $ pop : chr "Bangladesh" "Bangladesh" "Bangladesh" "Bangladesh" ...

*#select only taxonomies of a phylum for meta-analysis example (to save running time)*

tlm < -tabsex4$id[grep("k bacteria.p actinobacteria",tabsex4$id)] *#k\_\_bacteria.p\_\_fusobacteria*

metab.sex < -meta.taxa(taxcomdat=tabsex4[tabsex4$id %in% tlm,], summary.measure="RR",

pool.var="id", studylab="study", backtransform=FALSE, percent.meta=0.5, p.adjust.method="fdr")

*#show results by table #phylum level*

metatab.show(metatab=metab.sex$random,

com.pooled.tab=tabsex4[tabsex4$id %in% tlm,], tax.lev="l2",

showvar="genderMale", p.cutoff.type="p", p.cutoff=1, display="table")

## id estimate ll ul p

## 1 k bacteria.p actinobacteria 0.48 -0.27 1.23 0.208

## p.adjust

## 1 0.462

*#family level*

metatab.show(metatab=metab.sex$random,

com.pooled.tab=tabsex4[tabsex4$id %in% tlm,],

tax.lev="l5",

showvar="genderMale", p.cutoff.type="p",

p.cutoff=1,

display="table")

##

## 11 k bacteria.p actinobacteria.c coriobacteriia.o coriobacteriales. f coriobacteriaceae

## 10 k bacteria.p actinobacteria.c actinobacteria.o bifidobacteriales. f bifidobacteriaceae

## 9 k bacteria.p actinobacteria.c actinobacteria.o actinomycetales. f micrococcaceae

## 21 k bacteria.p actinobacteria.c actinobacteria.o bifidobacteriales. f bifidobacteriaceae.other

## 8 k bacteria.p actinobacteria.c actinobacteria.o actinomycetales. f corynebacteriaceae

## 7 k bacteria.p actinobacteria.c actinobacteria.o actinomycetales. f actinomycetaceae

## estimate ll ul p p.adjust

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | 11 | -0.26 -0.44 | -0.08 0.0049 | 0.0344 |
| ## | 10 | 0.52 -0.16 | 1.20 0.1345 | 0.4454 |
| ## | 9 | -0.09 -0.24 | 0.06 0.2198 | 0.4615 |
| ## | 21 | -0.12 -0.57 | 0.33 0.5925 | 0.8581 |
| ## | 8 | -0.02 -0.20 | 0.15 0.8026 | 0.8871 |
| ## | 7 | 0.02 -0.34 | 0.38 0.9209 | 0.9209 |

Now, we show meta-analysis results as a combined heatmap and forest plot. We use the function *metatab.show()* to get a result table then use *meta.niceplot()* to plot the result table.

metadat < -metatab.show(metatab=metab.sex$random,

com.pooled.tab=tabsex4, tax.lev="l5", showvar="genderMale", p.cutoff.type="p",

p.cutoff=1,

display="data")

meta.niceplot(metadat=metadat,sumtype="taxa",

level="main", p="p",

p.adjust="p.adjust",

phyla.col="rainbow",

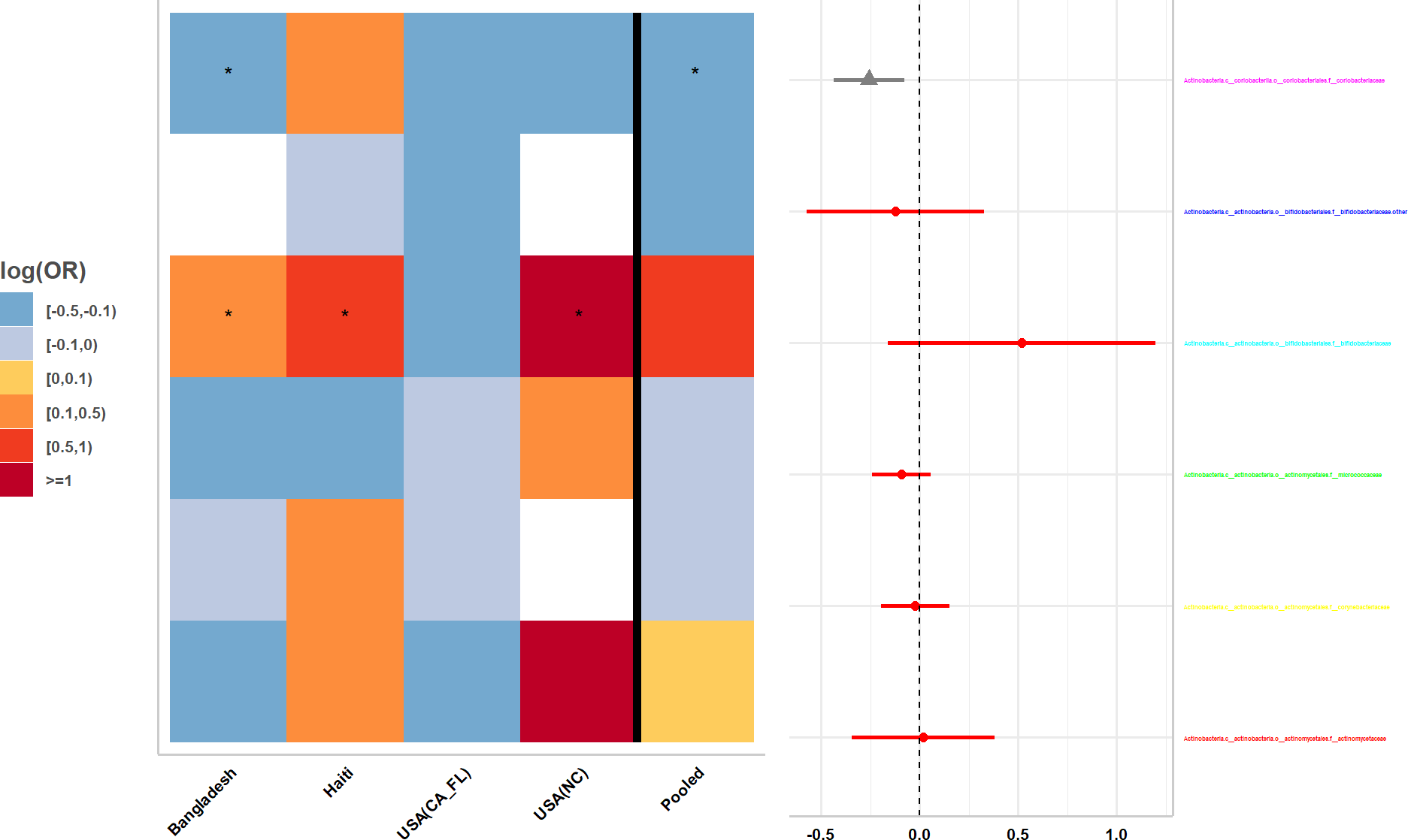
p.sig.heat="yes",

heat.forest.width.ratio =c(2.5, 2), leg.key.size=0.6,

leg.text.size=8, heat.text.x.size=8, heat.text.x.angle=45, forest.axis.text.y=3, forest.axis.text.x=8,

point.ratio = c(3,2),

line.ratio = c(1.5,1))



Source: Figure by author(s).

**Figure 113**. Metaanalysis nice plot.

#### Analysis and Meta-Analysis of Other Data

Meta-analysis may also be applied for other data such as alpha diversity indexes. For each study, the *alpha.compare()* function imports the outputs from *alpha\_rarefaction.py* QIIME1 script and calculates mean alpha diversity for different

indices for each sample based on a user defined rarefaction depth. Mean alpha diversity indexes are standardized to have a mean of 0 and standard deviation of 1 to make these measures comparable across studies. Standardized alpha diversity indexes are compared between groups adjusting for covariates using Linear model for cross sectional data and linear mixed effect model for longitudinal data. Meta-analysis across studies is then done and the results are displayed as a standard meta-analysis forest plot. The example below calculates mean alpha diversity indexes for a selected rarefaction depth, standardizes and compares standardized alpha diversity indexes between groups (male vs. female infants ≤ 6 months of age) adjusts for covariates (feeding status and infant age at sample collection) using the Bangladesh data [13].

data(alphadat) data(covar.rm)

covar.rm$sampleid < -tolower(covar.rm$sampleid)

alphacom < -alpha.compare(datlist=alphadat,

depth=3,

mapfile=covar.rm,

mapsampleid="sampleid",

comvar="gender", adjustvar=c("age.sample","bf"), longitudinal="yes",

age.limit=6,

standardize=TRUE)

alphacom$alphasum[,1:5]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ## | id | | Estimate.genderMale | | Std. | Error.genderMale |
| ## 1 | chao1 | | -0.0654 | |  | 0.150 |
| ## 2 | observed\_species | | -0.1428 | |  | 0.136 |
| ## 3 | pd\_whole\_tree | | -0.1325 | |  | 0.145 |
| ## 4 | shannon | | -0.4401 | |  | 0.199 |
| ## | t | value.genderMale | | Pr(>|t|).genderMale | | |
| ## 1 |  | -0.435 | | 0.6635 | | |
| ## 2 |  | -1.048 | | 0.2947 | | |
| ## 3 |  | -0.912 | | 0.3619 | | |
| ## 4 |  | -2.208 | | 0.0272 | | |

Now, we load save result outputs of the *alpha.compare()* function of the four study, and perform meta-analysis using the “*meta”*package [66] for an alpha diversity index (e.g. Shannon):

*# load saved results of 4 studies*

data(asum4)

asum4[,c(colnames(asum4)[1:5],"pop")]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ## | id | Estimate.genderMale | Std. | Error.genderMale |
| ## 1 | chao1 | 0.06527 |  | 0.0722 |
| ## 2 | observed\_species | 0.06529 |  | 0.0621 |
| ## 3 | pd\_whole\_tree | 0.03132 |  | 0.0502 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ## | 4 | shannon | -0.00128 |  | 0.0821 |
| ## | 5 | chao1 | -0.08862 |  | 0.3528 |
| ## | 6 | observed\_species | -0.09147 |  | 0.3453 |
| ## | 7 | pd\_whole\_tree | 0.04257 |  | 0.2952 |
| ## | 8 | shannon | -0.00911 |  | 0.2936 |
| ## | 9 | chao1 | -0.04503 |  | 0.1337 |
| ## | 10 | observed\_species | -0.06538 |  | 0.1145 |
| ## | 11 | pd\_whole\_tree | -0.15902 |  | 0.1076 |
| ## | 12 | shannon | -0.07053 |  | 0.1434 |
| ## | 13 | chao1 | 0.49658 |  | 0.5925 |
| ## | 14 | observed\_species | 0.23367 |  | 0.5294 |
| ## | 15 | pd\_whole\_tree | -0.01076 |  | 0.7331 |
| ## | 16 | shannon | 0.17831 |  | 0.5553 |
| ## |  | t value.genderMale | Pr(>|t|).genderMale | pop |  |
| ## | 1 | 0.9042 | 0.366 | Bangladesh |  |
| ## | 2 | 1.0510 | 0.293 | Bangladesh |  |
| ## | 3 | 0.6233 | 0.533 | Bangladesh |  |
| ## | 4 | -0.0155 | 0.988 | Bangladesh |  |
| ## | 5 | -0.2512 | 0.803 | Haiti |  |
| ## | 6 | -0.2649 | 0.792 | Haiti |  |
| ## | 7 | 0.1442 | 0.886 | Haiti |  |
| ## | 8 | -0.0310 | 0.975 | Haiti |  |
| ## | 9 | -0.3368 | 0.736 | USA(CA\_FL) |  |
| ## | 10 | -0.5710 | 0.568 | USA(CA\_FL) |  |
| ## | 11 | -1.4780 | 0.139 | USA(CA\_FL) |  |
| ## | 12 | -0.4920 | 0.623 | USA(CA\_FL) |  |
| ## | 13 | 0.8381 | 0.402 | USA(UNC) |  |
| ## | 14 | 0.4414 | 0.659 | USA(UNC) |  |
| ## | 15 | -0.0147 | 0.988 | USA(UNC) |  |
| ## | 16 | 0.3211 | 0.748 | USA(UNC) |  |

*# metaanalysis for Shannon index*

shannon.sex < - meta::metagen(Estimate.genderMale,

`Std. Error.genderMale`, studlab=pop, data=subset(asum4,id=="shannon"), sm="RD",

backtransf=FALSE)

*# Save current margin settings*

old\_mar < - par("mar")

*# Increase margins to avoid cropping*

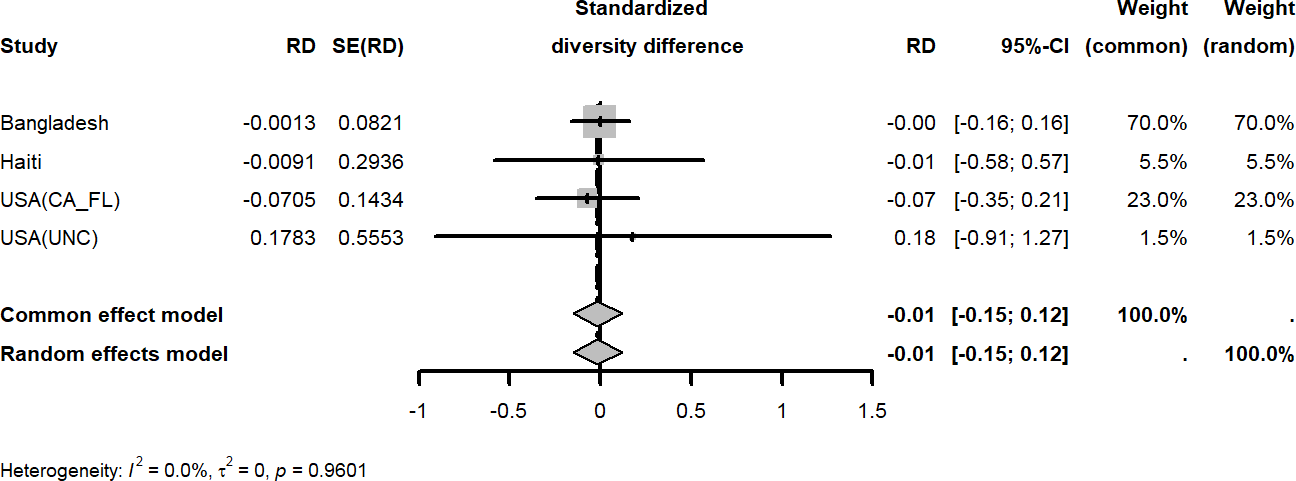
par(mar = c(5, 6, 4, 2))

meta::forest(shannon.sex,

smlab="Standardized \n diversity difference", sortvar=subset(asum4,id=="shannon")$pop, lwd=2,

xlim=c(-1, 1.5),

fontsize = 8)



Source: Figure by author(s).

**Figure 114**. Metaanalysis forest plot.

*# Reset margins to original settings*

par(mar = old\_mar)

shannon.sex

## Number of studies: k = 4

##

## RD 95%-CI z p-value ## Common effect model -0.0149 [-0.1495; 0.1198] -0.22 0.8288

## Random effects model -0.0149 [-0.1495; 0.1198] -0.22 0.8288 ##

## Quantifying heterogeneity (with 95%-CIs):

## tau^2 = 0 [0.0000; 0.0134]; tau = 0 [0.0000; 0.1157]

## I^2 = 0.0% [0.0%; 84.7%]; H = 1.00 [1.00; 2.56]

##

## Test of heterogeneity:

## Q d.f. p-value

## 0.30 3 0.9601

##

## Details of meta-analysis methods:

## - Inverse variance method

## - Restricted maximum-likelihood estimator for tau^2

## - Q-Profile method for confidence interval of tau^2 and tau

## - Calculation of I^2 based on Q

cbind(study=shannon.sex$studlab,pval=shannon.sex$pval)

## study pval

## [1,] "Bangladesh" "0.987599902164408"

## [2,] "Haiti" "0.9752390484969"

## [3,] "USA(CA\_FL)" "0.622739246734807"

## [4,] "USA(UNC)" "0.748126030769876"

Above are some examples illustrating the use of the R package “*metamicrobiomeR”* for the analysis and meta-analysis of microbiome data. This package only provides some limited functionalities for microbiome data analysis. Readers may explore other R packages for microbiome data for more comprehensive and elaborative analysis.

## Useful Resources

* The paper describing the **metamicrobiomeR** package on [BMC Bioinformatics](https://doi.org/10.1186/s12859-019-2744-2) [95].
* The **metamicrobiomeR** package hosted in [R cran](https://cran.r-project.org/web/packages/metamicrobiomeR/index.html) [4].
* An earlier version of the **metamicrobiomeR** package with full package tutorial on [Github](https://github.com/nhanhocu/metamicrobiomeR) (https://github.com/nhanhocu/metamicrobiomeR).

## Chapter Summary

The “*metamicrobiomeR*” package has the main functions below:

* *taxa.compare()* and *pathway.compare()* functions utilize Generalized Additive Models for Location, Scale and Shape with a zero-inflated beta family (GAMLSS-BEZI) for differential analysis of zero-inflated, compositional microbiome relative abundance data and produces standardized estimates (log odds ratios) suitable for meta-analysis
* Various options for data transformation are available
* *taxcomtab.show()* to display results *taxa.compare()* and *pathway.compare()*
* *meta.taxa()* function is used for meta-analysis of the effect estimates from multiple studies using random-effects or fixed-effects models.
* *metatab.show()* function to display results from the meta-analysis as tables
* *meta.niceplot()* function to visualize meta-analysis results as a combined heatmap and forest plot.
* *alpha.compare()* function to calculate, standardize, and compare alpha diversity indices between groups, with its outputs also being suitable for meta-analysis.
* *taxa.meansdn()* to summarize abundance data by group, *taxa.mean.plot()*

to visualize these means

# References

1. R Core Team. R: A language and environment for  statistical computing. R Foundation for Statistical  Computing, Vienna, Austria. 2025.

2. Posit team. RStudio: Integrated Development Environment for R [Internet]. Boston, MA; 2025. Available from: http://www.posit.co/

3. Xie Y, Allaire JJ, Grolemund G. R Markdown: The Definitive Guide. Boca Raton, Florida: Chapman and Hall/CRC; 2018.

4. Ho N. metamicrobiomeR: Microbiome Data Analysis & Meta-Analysis with GAMLSS-BEZI &  Random Effects. R package v 1.2. 2020.

5. Xie Y. Dynamic Documents with R and knitr. 2nd ed. Boca Raton, Florida: Chapman and Hall/CRC; 2015.

6. Allaire JJ, Xie Y, Dervieux C, McPherson J, Luraschi J, Ushey K, et al. rmarkdown: Dynamic Documents for R. R package v 2.29. 2024.

7. Galili T. installr: Using R to Install Stuff on Windows OS (Such As: R, “Rtools”, “RStudio”, “Git”, and More!). R package v 0.23.4. 2022.

8. Wickham H, Miller E, Smith D. haven: Import and Export “SPSS”, “Stata” and “SAS” Files. R package v 2.5.5. 2025.

9. Wickham H, Hester J, Bryan J. readr: Read Rectangular Text Data. R package v 2.1.5. 2024.

10. Garbett SP, Stephens J, Simonov K, Xie Y, Dong Z, Wickham H, et al. yaml: Methods to Convert R Data to YAML and Back. R package v 2.3.10. 2024.

11. Wickham H. tidyverse: Easily Install and Load the Tidyverse. 2023;

12. Zhu H. kableExtra: Construct Complex Table with kable and Pipe Syntax. R package v 1.4.0. 2024.

13. Subramanian S, Huq S, Yatsunenko T, Haque R, Mahfuz M, Alam MA, et al. Persistent gut microbiota immaturity in malnourished Bangladeshi children. Nature. 2014;510(7505):417–21.

14. Becker J, Chan C hong, Schoch D, Leeper TJ. rio: A Swiss-Army Knife for Data I/O. 2024.

15. Wickham H. httr: Tools for Working with URLs and HTTP. R package v 1.4.7. 2023.

16. Ooms J. The jsonlite Package: A Practical and Consistent Mapping Between JSON Data and R Objects. arXiv:14032805 [statCO]. 2014;

17. Wickham H, François R, Henry L, Müller K, Vaughan D. dplyr: A Grammar of Data Manipulation. R package v 1.1.4. 2023.

18. Wickham H. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York; 2016.

19. Heinzen E, Sinnwell J, Atkinson E, Gunderson T, Dougherty G. arsenal: An Arsenal of R Functions for Large-Scale Statistical  Summaries. R package v 3.6.3. 2021.

20. Kassambara A. ggpubr: ggplot2 Based Publication Ready Plots. R package v 0.6.1. 2025.

21. Aragón TJ. Population Health Data Science with R. 1st ed. Vol. 1. Berkeley, California: https://bookdown.org/taragonmd/phds/; 2019.

22. Rodrigues B. Modern R with the tidyverse. 1st ed. Vol. 1. https://modern-rstats.eu/; 2022.

23. Fox J, Weisberg S, Price B. car: Companion to Applied Regression. 3rd ed. Thousand Oaks: Sage; 2019.

24. Champely S. pwr: Basic Functions for Power Analysis. R package v 1.3-0. 2020.

25. Amala R, Kumarapandiyan G. samplesizeestimator: Calculate Sample Size for Various Scenarios. R package v 1.0.0. 2023.

26. Haynes AG, Lenz A, Stalder O, Limacher A. `presize`: An R-package for precision-based sample size calculation in clinical research. J Open Source Softw. 2021;6(60):3118.

27. Mathur SK. Statistical Bioinformatics with R. 1st ed. Elsevier; 2009.

28. Zar JH. Biostatistical Analysis. 5th ed. Prentice Hall; 2010.

29. Yoshida K, Bartel A. tableone: Create “Table 1” to Describe Baseline Characteristics with or without Propensity Score Weights. R package v 0.13.2. CRAN: Contributed Packages. 2014.

30. Sjoberg DD, Whiting K, Curry M, Lavery JA, Larmarange J. Reproducible Summary Tables with the gtsummary Package. R J. 2021;13(1):570–80.

31. Lemeshow S, Hosmer DW, Klar J, Lwanga SK. Adequacy of Sample Size in Health Studies. WORLD HEALTH ORGANIZATION; 1990.

32. Lenz A, Haynes AG, Limacher A. presize: Precision Based Sample Size Calculation. 2023.

33. Kuhn M, Vaughan D, Hvitfeldt E. yardstick: Tidy Characterizations of Model Performance. R package v 1.3.2. 2025.

34. Weisberg S. Applied Linear Regression . 4th ed. Wiley; 2014.

35. Zeileis A, Köll S, Graham N. Various Versatile Variances: An Object-Oriented Implementation of Clustered Covariances in R. J Stat Softw. 2020;95(1):1–36.

36. Pinheiro J, Bates D, R Core Team. nlme: Linear and Nonlinear Mixed Effects Models. R package v 3.1-168. 2025.

37. Bates D, Mächler M, Bolker B, Walker S. Fitting Linear Mixed-Effects Models Using lme4. J Stat Softw. 2015;67(1).

38. James G, Witten D, Hastie T, Tibshirani R. Introduction to Statistical Learning: with Applications in R. 2nd ed. Spinger; 2021.

39. Jackman S. pscl: Political Science Computational Laboratory. R package v 1.5.9. 2024.

40. Sing T, Sander O, Beerenwinkel N, Lengauer T. ROCR: visualizing classifier performance in R. Bioinformatics. 2005;21(20):7881.

41. Hilbe JM. Logistic Regression Models. 1st ed. Chapman & Hall; 2009.

42. Therneau TM, Grambsch PM. Modeling Survival Data: Extending the Cox Model. Springer; 2000.

43. Therneau TM. survival: Survival Analysis. R package v 3.8-3. 2024.

44. Kassambara A, Kosinski M, Biecek P. survminer: Drawing Survival Curves using ggplot2. R package v 0.5.0. 2024.

45. Grolemund G, Wickham H. Dates and Times Made Easy with lubridate. J Stat Softw. 2011;40(3):1–25.

46. Friedman J, Hastie T, Tibshirani R, Narasimhan B, Tay K, Simon N, et al. glmnet: Lasso and Elastic-Net Regularized Generalized Linear Models. 2025.

47. Gray B. cmprsk: Subdistribution Analysis of Competing Risks [Internet]. 2024. Available from: https://www.R-project.org

48. Moore DF. Applied Survival Analysis Using R. Springer Cham; 2016.

49. Fine JP, Gray RJ. A Proportional Hazards Model for the Subdistribution of a Competing Risk. J Am Stat Assoc. 1999 Jun;94(446):496–509.

50. Crowley J, Hu M. Covariance Analysis of Heart Transplant Survival Data. J Am Stat Assoc. 1977 Mar;72(357):27.

51. Halekoh U, Højsgaard S, Yan J. The R Package geepack for Generalized Estimating Equations. J Stat Softw. 2006;15/2:1–11.

52. Wood S. mgcv: Mixed GAM Computation Vehicle with Automatic Smoothness  Estimation. R package v 1.9-3. 2025.

53. Carey VJ. gee: Generalized Estimation Equation Solver. R package v 4.13-29. 2024.

54. Lüdecke D. sjPlot: Data Visualization for Statistics in Social Science. R package v2.9.0. 2025.

55. Lüdecke D. sjlabelled: Labelled Data Utility Functions [Internet]. R package v 1.2.0. 2022. Available from: https://strengejacke.github.io/sjlabelled/

56. Lüdecke D. sjmisc: Data and Variable Transformation Functions. R package v 2.8.11. 2025.

57. Ripley B, Venables B. MASS: Support Functions and Datasets for Venables and Ripley’s MASS. R package v 7.3-65. 2025.

58. Singer JD, Willett JB. Applied Longitudinal Data Analysis. Oxford University PressNew York; 2003.

59. Galecki A, Burzykowski T. Linear Mixed-Effects Models Using R. New York, NY: Springer New York; 2013.

60. Hubbard AE, Ahern J, Fleischer NL, Laan M Van der, Lippman SA, Jewell N, et al. To GEE or Not to GEE. Epidemiology. 2010 Jul;21(4):467–74.

61. Ho NT, Abe SK, Rahman MdS, Islam R, Saito E, Gupta PC, et al. Diabetes is associated with increased liver cancer incidence and mortality in adults: A report from Asia Cohort Consortium. Int J Cancer. 2024 Sep 25;155(5):854–70.

62. Auguie B. gridExtra: Miscellaneous Functions for “Grid” Graphics. R package v 2.3. 2017.

63. Harrell Jr FE. Hmisc: Harrell Miscellaneous. R package v 5.2-3. 2025.

64. Revelle W. psych: Procedures for Psychological, Psychometric, and Personality  Research. R package v 2.5.6. 2025.

65. Warnes GR, Bolker B, Lumley T, Johnson RC, Jain N, Schwartz M, et al. gmodels: Various R Programming Tools for Model Fitting. R package v 2.19.1. 2024.

66. Schwarzer G. meta: General Package for Meta-Analysis [Internet]. 2025. Available from: https://github.com/guido-s/meta/

67. Pedersen TL. patchwork: The Composer of Plots [Internet]. 2025. Available from: https://patchwork.data-imaginist.com

68. De Onis M, Onyango AW, Borghi E, Siyam A, Nishida C, Siekmann J. Development of a WHO growth reference for school-aged children and adolescents. Bull World Health Organ. 2007;85(9).

69. De Onis M. WHO Child Growth Standards based on length/height, weight and age. Acta Paediatrica, International Journal of Paediatrics. 2006;95(SUPPL. 450).

70. Schumacher D. anthro: Computation of the WHO Child Growth Standards. R package v 1.0.1. 2023.

71. Myatt M, Guevarra E. zscorer: Child Anthropometry z-Score Calculator. R package v 0.3.1. 2019.

72. Signorell A. DescTools: Tools for Descriptive Statistics. R package v 0.99.60. 2025.

73. Mayer M. confintr: Confidence Intervals. R package v 1.0.2. 2023.

74. Stasinopoulos M, Rigby R. gamlss: Generalized Additive Models for Location Scale and Shape. R package v 5.4-22. 2024.

75. Rudis B. hrbrthemes: Additional Themes, Theme Components and Utilities for ggplot2. R package v 0.8.7. 2024.

76. Garnier S. viridis: Colorblind-Friendly Color Maps for R. R package v 0.6.5. 2024.

77. Lenth R V. emmeans: Estimated Marginal Means, aka Least-Squares Means. R package v 1.11.2. 2025.

78. Aphalo PJ. ggpmisc: Miscellaneous Extensions to ggplot2. R package v 0.6.2. 2025.

79. Wickham H. stringr: Simple, Consistent Wrappers for Common String Operations. R package v 1.5.1. 2023.

80. Cole TJ. The LMS method for constructing normalized growth standards. Eur J Clin Nutr. 1990;44(1).

81. Ho NT, Bangsberg D, Hermiston ML, Do CT, Phung LN, Nguyen Q Van, et al. Overweight & obesity epidemic, temporal trends and regional disparities in physical growth of Vietnamese children. Sci Rep. 2026;

82. Qiu W, Chavarro J, Lazarus R, Rosner B, Ma J. powerSurvEpi: Power and Sample Size Calculation for Survival Analysis of  Epidemiological Studies. R package v 0.1.5. 2025.

83. G.Zhang EZ; VQW; SCC; H. TrialSize: R Functions for Chapter 3,4,6,7,9,10,11,12,14,15 of Sample Size  Calculation in Clinical Research. R package v 1.4.1. 2024.

84. Qi H, Zhu F. SampleSize4ClinicalTrials: Sample Size Calculation for the Comparison of Means or  Proportions in Phase III Clinical Trials. R package v 0.2.3. 2021.

85. Anderson K. gsDesign: Group Sequential Design. R package v 3.6.9. 2025.

86. Snow G. blockrand: Randomization for Block Random Clinical Trials. R package v 1.5. 2020.

87. Uschner D, Schindler D, Hilgers RD, Heussen N. randomizeR: An R Package for the Assessment and Implementation of Randomization in Clinical Trials. J Stat Softw. 2018;85(8):1–22.

88. Chow SC, Shao J, Wang H, Lokhnygina Y. Sample Size Calculations in Clinical Research: Third Edition. 3rd ed. Third edition. | Boca Raton : Taylor & Francis, 2017. | Series: Chapman & Hall/CRC biostatistics series | “A CRC title, part of the Taylor & Francis imprint, a member of the Taylor & Francis Group, the academic division of T&F Informa plc.”: Chapman and Hall/CRC; 2017.

89. Arjyal A, Basnyat B, Nhan HT, Koirala S, Giri A, Joshi N, et al. Gatifloxacin versus ceftriaxone for uncomplicated enteric fever in Nepal: an open-label, two-centre, randomised controlled trial. Lancet Infect Dis. 2016 May;16(5):535–45.

90. Le T, Kinh N Van, Cuc NTK, Tung NLN, Lam NT, Thuy PTT, et al. A Trial of Itraconazole or Amphotericin B for HIV-Associated Talaromycosis. New England Journal of Medicine. 2017 Jun 15;376(24):2329–40.

91. Ho NT, Hughes SG, Ta VT, Phan LT, Đỗ Q, Nguyễn TV, et al. Safety, immunogenicity and efficacy of the self-amplifying mRNA ARCT-154 COVID-19 vaccine: pooled phase 1, 2, 3a and 3b randomized, controlled trials. Nat Commun. 2024 May 14;15(1):4081.

92. Zhang Y, Xiao N, Anderson K, Zhu Y. R for Clinical Study Reports and Submission. https://r4csr.org/; 2025.

93. Friedman LM, Furberg CD, DeMets DL, Reboussin DM, Granger CB. Fundamentals of Clinical Trials. 5th ed. Springer Nature; 2015.

94. Daniels B, Coutsoudis A, Moodley-Govender E, Mulol H, Spooner E, Kiepiela P, et al. Effect of co-trimoxazole prophylaxis on morbidity and mortality of HIV-exposed, HIV-uninfected infants in South Africa: a randomised controlled, non-inferiority trial. Lancet Glob Health. 2019 Dec;7(12):e1717–27.

95. Ho NT, Li F, Wang S, Kuhn L. metamicrobiomeR: an R package for analysis of microbiome relative abundance data using zero-inflated beta GAMLSS and meta-analysis across studies using random effects models. BMC Bioinformatics. 2019 Dec 16;20(1):188.

96. Rigby RA, Stasinopoulos DM. Generalized additive models for location, scale and shape. Journal of the Royal Statistical Society. 2005 Jun 1;54(3):507–54.

97. Chen EZ, Li H. A two-part mixed-effects model for analyzing longitudinal microbiome compositional data. Bioinformatics. 2016 Sep 1;32(17):2611–7.

98. McMurdie PJ, Holmes S. Waste Not, Want Not: Why Rarefying Microbiome Data Is Inadmissible. PLoS Comput Biol. 2014 Apr 3;10(4):e1003531.

99. Min Y, Agresti A. Random effect models for repeated measures of zero-inflated count data. Stat Modelling. 2005;(5):1–19.

100. Aitchison J, Barceló-Vidal C, Martín-Fernández JA, Pawlowsky-Glahn V. Logratio Analysis and Compositional Distance. Math Geol. 2000;32(3).

101. Palarea-Albaladejo J, Martin-Fernandez JA. zCompositions -- R package for multivariate imputation of left-censored data under a compositional approach. Chemom Intell Lab Syst. 2015;(143):85–96.

102. Chen L, Reeve J, Zhang L, Huang S, Wang X, Chen J. GMPR: A robust normalization method for zero-inflated count data with application to microbiome sequencing data. PeerJ. 2018 Apr 2;6:e4600.

103. Bender JM, Li F, Martelly S, Byrt E, Rouzier V, Leo M, et al. Maternal HIV infection influences the microbiome of HIV-uninfected infants. Sci Transl Med. 2016 Jul 27;8(349):349ra100.

104. Pannaraj PS, Li F, Cerini C, Bender JM, Yang S, Rollie A, et al. Association Between Breast Milk Bacterial Communities and Establishment and Development of the Infant Gut Microbiome. JAMA Pediatr. 2017;90095(7):647–54.

105. Thompson AL, Monteagudo-Mera A, Cadenas MB, Lampl ML, Azcarate-Peril MA. Milk- and solid-feeding practices and daycare attendance are associated with differences in bacterial diversity, predominant communities, and metabolic and immune function of the infant gut microbiome. Front Cell Infect Microbiol. 2015;5:3.

106. Warnes GR, Gorjanc G, Magnusson A, Andronic L, Rogers J, MacQueen D, et al. gdata: Various R Programming Tools for Data Manipulation. R package v 3.0.1. 2005.

107. Caporaso JG, Kuczynski J, Stombaugh J, Bittinger K, Bushman FD, Costello EK, et al. QIIME allows analysis of high-throughput community sequencing data. Nat Methods. 2010 May 11;7(5):335–6.

108. Langille MGI, Zaneveld J, Caporaso JG, McDonald D, Knights D, Reyes JA, et al. Predictive functional profiling of microbial communities using 16S rRNA marker gene sequences. Nat Biotechnol. 2013 Aug 25;31(9):814–21.

109. Morgan XC, Tickle TL, Sokol H, Gevers D, Devaney KL, Ward D V, et al. Dysfunction of the intestinal microbiome in inflammatory bowel disease and treatment. Genome Biol. 2012 Sep 26;13(9):R79.

# Index

≤

≤ 5 years old, 218

≥

≥ 5 years old, 218

9

95%CI, 206, 209, 218, 238

A

accumulated risk, 161

Accuracy, 129

Additivity, 152, 153

Adjusted R-squared, 109

*adjustvar*, 285

adverse events, 247, 276

AIC, 109, 128, 134, 158

*AIC()*, 134, 158

Akaike Information Criterion, 109, 128, 134, 158

alpha diversity, 294

*alpha.compare()*, 283, 294

*anova()*, 150, 158, 188

*anthro*, 203, 224

*anthro\_prevalence()*, 227

*anthro\_zscores()*, 224

anthropometric, 203

Area Under the Curve, 132

array, 26

*arsenal*, 49, 67, 74, 115, 202, 203, 205, 247, 260, 261

as.Date(), 142

assumption, 71, 152

AUC, 132

Auto-regressive, 176

B

backward, 109, 159

barplot, 37

Base R, 23

baseline hazard function, 149

baseline summaries, 247

Bayesian Information Criterion, 109, 134, 158

Between-Subject Variance, 185

bias-corrected and accelerated, 209

BIC, 109, 134, 158

*BIC()*, 134, 158

binary, 115

Binary Outcome, 181, 248

binomial, 67, 115, 126, 177, 195

binomial distribution, 68

Blocked Randomization, 255

*blockrand*, 247, 255

bootstrap confidence interval, 209

Box–Cox transformation, 243

boxplot, 43

C

*c()*, 31

*car*, 67

case study, 247

*case\_when()*, 222

categorical, 115

Categorical predictor, 92

categorical variables, 73

Censoring, 141

Centered Log Ratio, 282

*centiles()*, 244

*centiles.fan()*, 244

Central Limit Theorem, 67

child growth, 203

child growth reference data, 203

*chisq.test()*, 73

Chi-square test, 73

*ci\_median\_diff()*, 203, 209

CIF, 163

classification measures, 129

Classification Metrics, 128

*cmprsk*, 142, 164, 201

coef(), 84

coefficients, 91, 117, 119, 128, 149, 190

color code, 217

competing risk, 162

Competing Risk Analysis, 141

*comvar*, 285

Concordance Index, 155

confidence interval, 206

confint(), 84, 85

*confintr*, 203, 209

confusion matrix, 128

Continuous Outcome, 248

Continuous variables, 70

Cook’s distance, 91

Correlation, 175

correlation model, 176

correlation structure, 176

*corstr*, 177

Count Outcome, 182

Covariance, 175

covariates, 149

Cox model, 182

Cox proportional hazards model, 141, 149

*cox.zph()*, 144, 153

*coxph()*, 144, 149

*cuminc()*, 163

Cumulative Hazard Function, 161

Cumulative Incidence Function, 161

cumulative incidence plots, 269

cumulative probability, 161

*cut()*, 227

*cv.glmnet()*, 160

D

data frame, 27

Data Manipulation, 34

data object, 29

Data visualization, 37

*data.frame()*, 31

dataset, 224

dates, 142

*dbinom()*, 68

density, 39

Dependence, 175

dependency, 89

*DescTools*, 203

Deviance, 128

Deviance Residuals, 127, 156

*dfbeta*, 91

diagnostic measures, 91

diagnostic plot, 88

difference, 209

differential, 282

*difftime()*, 142

Discrimination power, 132

distribution, 67

*dnorm*, 67

*dplyr*, 32, 34, 45

E

Each Study, 188

effect size, 78, 248

efficacy, 247

*emmeans*, 203

Entering Data, 31

Equal Variance Assumption, 88

Error rate, 129

errors, 89

*ES.h()*, 80, 249

event, 141

Exchangeable, 176

Exporting Data, 34

Extract effects, 189

Extreme tallness, 240

F

*facet\_grid()*, 64

False Negative, 129

False Positive, 129

*family*, 126, 177, 194, 285

figures, 247

*fill*, 38

filter, 282

*filter()*, 34

Fine-Gray Model, 164

*fisher.test()*, 74

Fisher’s Exact Test, 74

fixed effects, 169, 184

forest plot, 193, 291

*forest()*, 193

forward, 109, 159

function, 29, 184

functional pathway, 284

G

*gamlss*, 203, 243, 282

GAMLSS, 281

GAMLSS-BEZI, 281

*gaussian*, 177

Gaussian, 194

*gdata*, 283

*gee*, 169

GEE, 169

GEE model, 175

*geeglm()*, 177, 181, 210

*geepack*, 169, 177, 203, 210

Generalized Additive Models for Location, Scale, and Shape, 203

Generalized Estimating Equations, 169, 210

Generalized Linear Mixed-Effects Models, 169, 185

Generalized Linear Models, 91, 115, 117

genus, 287

*geom = "table"*, 214

*geom\_bar()*, 37

*geom\_boxplot()*, 43

*geom\_density()*, 39

*geom\_dotplot()*, 44

*geom\_histogram()*, 38

*geom\_point()*, 42

Geometric Mean of Pairwise Ratios, 282

*ggarrange()*, 65, 216

*ggcompetingrisks()*, 163

*ggplot()*, 97

*ggplot2*, 34, 37, 169, 203, 206

*ggpmisc*, 203

*ggpubr*, 65, 203

*ggsurvplot()*, 180, 269

GitHub API, 32

Glaz & Sison’s method, 223

*glm()*, 91, 115, 119, 126

*glmer()*, 194

GLMM, 169

*glmnet*, 142, 160

*gmodels*, 202

Goodness-of-Fit, 156

*gridExtra*, 201, 203

*group\_by()*, 45, 46, 273

growth curves, 203, 244

growth trajectory, 220

*gsDesign*, 247

*gtsummary*, 202

H

*haven*, 25

Hazard Function, 141, 142

hazard rate, 149

hazard ratio, 141, 149, 151

heatmap, 291

help for package, 25

heterogeneity, 184

Heterogeneity Test, 184

heteroscedasticity, 88

histogram, 38

*Hmisc*, 201

Homogeneity, 71

Homoscedasticity, 87, 88

HR, 149

*hrbrthemes*, 203

*httr*, 32

I

*id*, 177

*if*, 190

*import()*, 32

*import\_list()*, 32

Importing Data, 31

incidence rate, 178

Independence, 70, 89, 176

independent, 87

individual, 226

Individual Data, 220

*influence.measures()*, 91

Influential observations, 91

Install, 24

*install.packages()*, 24

*installr*, 25

instantaneous rate, 151

interaction, 106, 153

intercept, 85

interpretation, 108, 127

interquartile, 206

interquartile range, 72

*interval()*, 142

*iqr()*, 206

J

*jsonlite*, 32

K

*kableExtra*, 25, 67, 115, 142, 169, 202, 203, 247

Kaplan-Meier, 141, 146

Kaplan-Meier plot, 180

Kaplan-Meir Curve, 188

KEGG, 284

*knitr*, 25, 67, 115, 142, 169, 202, 203, 247

Kruskal Wallis’ test, 73

*kruskal.test()*, 73

L

Lambda, 243

Lambda-Mu-Sigma, 203

*lapply()*, 25

LASSO, 160

Likelihood ratio test, 150, 158, 188

linear assumption, 87

Linear mixed-effects models, 184

Linear Regression line, 117

Linear relationship, 87

Linearity, 152, 154

*link*, 194

list, 27

*lm()*, 84

*lme4*, 169, 185

*lmer()*, 185

*lmerTest*, 186

LMS, 203, 243

*lms()*, 243

Load, 24

log (odds ratio), 281

log(odds), 119, 120

Logistic function, 117

Logistic Regression, 115

Logistic Regression curve, 117

log-likelihood, 122

log-rank test, 148

*longitudinal*, 286

longitudinal correlation, 176

Longitudinal data, 169

loop, 189, 188, 209

*lstrends()*, 212

*lubridate*, 142, 201

M

MaAsLin, 287

mae(), 112

Managing Biomedical Data, 30

Mantel-Haenszel test, 148

*mapfile*, 288

Marginal Effects, 190

Martingale Residuals, 156

*MASS*, 169, 203

matrix, 26

Maximum Likelihood, 120

Maximum Likelihood Estimation, 122

McFadden’s Pseudo R-Squared, 134

McFadden’s R-squared, 122

mean, 72, 80

Mean Absolute Error, 111

mean model, 176

Mean Squared Error, 111

*mean.filter*, 284

*medci()*, 206, 218

median, 72, 206, 218

Median survival time, 145

*MedianCI()*, 203

medians, 209

*meta*, 202, 193, 295

*meta.niceplot()*, 283, 294

*meta.taxa()*, 282, 283, 290

meta-analysis, 193, 281, 290

*metagen()*, 193

*metamicrobiomeR*, 24, 281

*metatab.show()*, 283, 291

*mgcv*, 169

*microbiomeage()*, 283

Misclassification, 129

Model Accuracy, 155

Model calibration, 112

model complexity, 109

Model Diagnostics, 190

Model Evaluation, 128

model fit, 86, 109

model fit statistics, 134

Model selection, 109, 158

model validation, 110, 135, 141

Mu, 243

*MultinomCI()*, 223

Multiple linear regression, 103

Multiple testing, 282

multivariable, 119, 151

*mutate()*, 36

N

*n\_distinct()*, 276

Negative, 129

Nelson-Aalen Estimator, 161

new function, 184, 206, 232

non-inferiority, 247

Non-inferiority trial, 250

non-informative censoring, 149

non-linear term, 87

non-linear transformations, 87

non-parametric tests, 70

normal, 67

Normal BMI, 241

normal distribution, 67

Normal HFA, 240

Normal WFH, 238

normality, 91

normally distributed, 87

*nSurv()*, 253

nuisance, 176

Null deviance, 128

number at risk, 145

number of events, 145

nutritional status classification, 203, 217, 222

O

obesity, 224, 238

*offset*, 177

operation, 30

outcome, 87, 115, 247

overweight, 224, 238

P

*p.adjust.method*, 286

Parameter, 80

parametric tests, 70

*patchwork*, 202, 203

*pathsum*, 288

*pathtab*, 288

Pathway, 288, 290

*pathway.compare()*, 283, 285, 288

*pbinom()*, 69

Penalized Regression, 160

*percent.filter*, 286

*percent.meta*, 291

Performance Metrics, 111

*performance()*, 129

Peto-Wilcoxon test, 148

phylum, 284

pipe operator, 34

*pivot\_longer()*, 273

*pivot\_wider()*, 273

plot, 190, 284

plot(), 84, 129

*plot\_model()*, 190, 215

*pnorm*, 67

*poisson*, 177

Poisson, 197

pooled estimates, 291

population-average, 169

Positive, 129

Positive predictive value, 129

power, 78, 247, 248

*powerSurvEpi*, 247, 250

*pR2()*, 123, 134

*prec\_prop()*, 81

Precision, 80, 129

predict(), 84, 97, 129

predicted values, 86

Prediction, 128

*prediction()*, 129

predictor, 87

Preferred Term, 276

*presize*, 67, 78, 81

prevalence, 203, 223, 238

probability, 69, 115

*propmed.rel*, 285

proportional hazards, 149, 152

Proportional Hazards Assumption, 141

*pscl*, 115, 123, 134

*pspline()*, 154

*psych*, 202

publication ready table, 74

p-value, 149

*pwr*, 67, 78, 247, 248

*pwr.2p.test()*, 80, 249

*pwr.t.test()*, 79, 248

Q

*QIC()*, 180, 210

*qnorm*, 67

Q-Q plot, 70, 89

*qqline()*, 70

*qqnorm()*, 70

quantiles, 206

Quasi Information Criterion, 180, 210

R

R, 23

R Markdown, 23

R Packages, 24

*R*2, 111, 122

random effects, 169, 184

random intercept, 185, 282

random slope, 186

randomization, 247

*randomizeR*, 247

*read.multi()*, 283

*readjust.p*, 286

*readr*, 25

Receiver Operating Characteristics Curve, 132

*relabund.filter*, 286

relative abundance, 281

*rename()*, 36

*rename\_at()*, 34

repeated measurements, 169

reporting-ready, 247

Residual deviance, 128

residuals, 85, 156, 193

residuals(), 84, 156

*rio*, 32, 169, 202, 203, 247

*rmarkdown*, 25, 67, 115, 202, 203

*rmse()*, 112

*rnorm*, 67

ROC, 132

*ROCR*, 115

Root Mean Squared Error, 111

*rsq()*, 112

R-squared, 111, 122

RStudio, 23

S

sample size, 78, 247

*sample\_frac*(), 135

*SampleSize4ClinicalTrials*, 247, 252

*samplesizeestimator*, 67, 78, 80

scatter plot, 42

Score test, 148, 158

SD, 218

*select()*, 34

Sensitivity, 129

sensitivity analysis, 195

Serious Adverse Events, 273

*show.taxname*, 284

Sigma, 243

Sigmoid function, 117

significance level, 78, 248

Simple linear regression, 83

Simple Randomization, 254

*sjlabelled*, 169, 190

*sjmisc*, 169, 190

*sjPlot*, 169, 190, 203, 215

slope, 85

smooth function, 243

smoothed line, 42

smoothed percentile curves, 244

spaghetti plot, 170

Specificity, 129

standard deviation, 218

standard error, 88, 149, 180

Standardized, 295

standing, 232

start-stop format, 165

*stat\_smooth*, 42

*stat\_summary()*, 207

*step()*, 109, 159

stepwise, 109, 159

*str()*, 28

*strata*, 265

*strata()*, 152

Stratified Cox model, 152

Stratified Randomization, 257

*stringr*, 203

stunting, 224, 240

Subdistribution Hazard Model, 164

Subgroup Analysis, 184

*subhe()*, 184

subject-specific, 169

*summarise()*, 45, 273

Summary Statistics, 45

summary table, 49, 202

summary(), 84, 85, 149

*summary.measure*, 290

superiority, 247

Superiority trials, 248

supine length, 232

*Surv()*, 143

*surv\_fit()*, 269

survdiff(), 144

*survfit()*, 144, 145, 147

*survival*, 141, 142, 201, 247, 260

Survival data, 141

Survival Function, 141

survival outcome, 149

survival probability, 145

survival time, 141

*survminer*, 141, 142, 201, 260

*Survreg()*, 144

System Organ Class, 276

T

*t.test()*, 71

*tableby()*, 49, 76, 202, 261, 265

*tableby.control()*, 52, 76, 261

tables, 247

*tax.lev*, 285

*taxa.compare()*, 281, 285

*taxa.filter()*, 283, 284

*taxa.mean.plot()*, 283, 284

*taxa.meansdn()*, 283, 284

*taxcomtab.show()*, 283, 286

Taxonomic, 285

Taxonomy, 290

*taxtab*, 285

Temporal trend, 210

*termplot()*, 154

test, 135

Testing Set, 110

thinness, 224, 241

*tidyverse*, 25, 34, 115, 142, 169, 202, 203, 222, 247, 260

Time dependent covariates, 141, 165

time-to-event, 141

Time-to-Event Outcome, 250

training, 135

Training Set, 110

transform, 86, 286

transformation, 91, 161, 282

trend, 87

trend slopes, 215

*TrialSize*, 247, 251

true negatives, 129

true positive, 129

T-test, 70

*TwoSampleMean.NIS()*, 251

*TwoSampleProportion.NIS()*, 251

*type*, 190

U

Unequal variance, 88

Univariable Cox Model, 188

Unstructured, 176

Update, 24

updating packages, 25

V

vaccine trial, 260

validation, 195

variance, 71, 175

vector, 25

*viridis*, 203

Visualization, 190

W

Wald test, 158

wasting, 224, 238

*wgsData*, 230

WHO, 203

WHO 2007 Growth References, 217

WHO median, 218

WHO reference data, 222

*wilcox.test()*, 72

Wilcoxon’s rank sum test, 72

Within-Subject Variance, 185

World Health Organization, 203

*wrap\_plots()*, 188

writing a loop, 189

Y

*yaml*, 25

Z

*zcal()*, 232

zero-inflated beta, 281

*zeroreplace.method*, 286

zero-replacement, 282

zscorer, 203, 230

Z-scores, 203, 224

z-statistic, 149