Introduction to Basic Regression with R

Ashley I Naimi

September 2024

Contents

1 Regression in R

In this section, we will look at several ways to fit regression models with the goal of estimating an exposure-outcome association adjusting for several potential confounding variables. R offers a great degree of flexibility in fitting models to data (again, there are many ways to do the same thing in R.) This section will seek to provide a way forward, and demonstrate how to use regression modeling in different ways.

Suppose we wanted to use the NHEFS data to estimate the confounder adjusted effect of quitting smoking on weight change (continuous) and death (binary). We can do this using the analytic dataset we created in the previous section. First, we'll load the relevant libraries needed to conduct our analysis.

```
pacman::p_load(tidyverse,
               here,
               broom,
               boot)
```

In the packages above, we've already been introduced to elements of the `tidyverse` and the `here` package. The `broom` package offers tools to extract information from generalized linear models, and create datsets with them. In effect, it allows us to extract regression information in a neat and tidy way. Finally, the `boot` package allows us to implement the bootstrap when needed.[1]

[1] Though there are many ways to implement the bootstrap, and we need not always use the `boot` package to do this.

First, we'll load the relevant analytic dataset that we created in the previous section:

```
nhefs <- read_csv(here("data","analytic_data.csv"))

dim(nhefs)
```

```
## [1] 1476    9
```

```
nhefs %>% print(n=5)
```

```
## # A tibble: 1,476 x 9
##    seqn  qsmk   sex   age  race income wt82_71 death   map
```

```
##   <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>   <dbl> <dbl> <dbl>
## 1   233     0     0    42     1     19  -10.1      0 122.
## 2   235     0     0    36     0     18    2.60     0  94.3
## 3   244     0     1    56     1     15    9.41     0  88.3
## 4   245     0     0    68     1     15    4.99     1 101.
## 5   252     0     0    40     0     18    4.99     0  90.7
## # i 1,471 more rows
```

2 Conditionally Adjusted Reegression Model for the Mean Difference

2.1 A Basic Model

We'll start with a regression model that allows us to estimate the association between quitting smoking and weight change:

```
#' Here, we start fitting relevant regression models to the data.

#' This model can be used to quantify a conditionally adjusted
#' mean difference with correct standard error
model_MD <- glm(wt82_71 ~ qsmk + sex + age + race + income + map,
                data = nhefs,
                family = gaussian("identity"))

#' summary() is a base R function that reports the fit of a
#'  regression model neatly in the console
summary(model_MD)
```

```
##
## Call:
## glm(formula = wt82_71 ~ qsmk + sex + age + race + income + map,
##     family = gaussian("identity"), data = nhefs)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.91357    2.34139   0.817    0.414
```

```
## qsmk         2.83202    0.45922   6.167 8.98e-10 ***
## sex         -0.06788    0.39735  -0.171    0.864
## age         -0.17504    0.01694 -10.331  < 2e-16 ***
## race        -0.26537    0.60902  -0.436    0.663
## income       0.05549    0.07769   0.714    0.475
## map          0.07065    0.01751   4.034 5.76e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 56.07977)
##
##     Null deviance: 90783  on 1475  degrees of freedom
## Residual deviance: 82381  on 1469  degrees of freedom
## AIC: 10141
##
## Number of Fisher Scoring iterations: 2
```

```
#' tidy() is a broom function that output the fit of a
#' regression model as a tidy dataset
tidy(model_MD)
```

```
## # A tibble: 7 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   1.91      2.34       0.817 4.14e- 1
## 2 qsmk          2.83      0.459      6.17  8.98e-10
## 3 sex          -0.0679    0.397     -0.171 8.64e- 1
## 4 age          -0.175     0.0169   -10.3   3.37e-24
## 5 race         -0.265     0.609     -0.436 6.63e- 1
## 6 income        0.0555    0.0777     0.714 4.75e- 1
## 7 map           0.0707    0.0175     4.03  5.76e- 5
```

2.2 Including Interactions

With this model, we may want to include an interaction. We can do this in two ways in R:

```r
#' interaction: method 1 -- colon includes only interaction
model_MD <- glm(wt82_71 ~ qsmk + sex + age + sex:age + race + income + map,
                data = nhefs,
                family = gaussian("identity"))

summary(model_MD)$coefficients
```

```
##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  0.85937943 2.49148337  0.3449268 7.301988e-01
## qsmk         2.82978261 0.45913855  6.1632433 9.189754e-10
## sex          1.69479607 1.48030386  1.1448974 2.524383e-01
## age         -0.15521810 0.02332453 -6.6547141 3.996491e-11
## race        -0.30463248 0.60974165 -0.4996091 6.174252e-01
## income       0.05574146 0.07768112  0.7175677 4.731381e-01
## map          0.07250862 0.01757438  4.1258146 3.901631e-05
## sex:age     -0.04035235 0.03264493 -1.2360985 2.166197e-01
```

```r
#' interaction: method 2 -- asterisk includes main effect and interaction
model_MD <- glm(wt82_71 ~ qsmk + sex*age + race + income + map,
                data = nhefs,
                family = gaussian("identity"))

summary(model_MD)$coefficients
```

```
##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  0.85937943 2.49148337  0.3449268 7.301988e-01
## qsmk         2.82978261 0.45913855  6.1632433 9.189754e-10
## sex          1.69479607 1.48030386  1.1448974 2.524383e-01
## age         -0.15521810 0.02332453 -6.6547141 3.996491e-11
## race        -0.30463248 0.60974165 -0.4996091 6.174252e-01
## income       0.05574146 0.07768112  0.7175677 4.731381e-01
## map          0.07250862 0.01757438  4.1258146 3.901631e-05
## sex:age     -0.04035235 0.03264493 -1.2360985 2.166197e-01
```

2.3 Including Splines

The word spline is an engineering/architectural term. It refers to a flexible piece of material that individuals would use to draw up blue-prints that incorporated flexible curves:

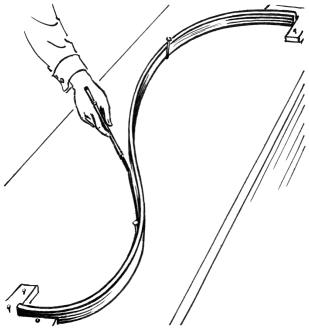


Figure 1: An illustration of a engineering/architectural spline use to draw flexible curves for blueprint diagrams. Source: Wikipedia

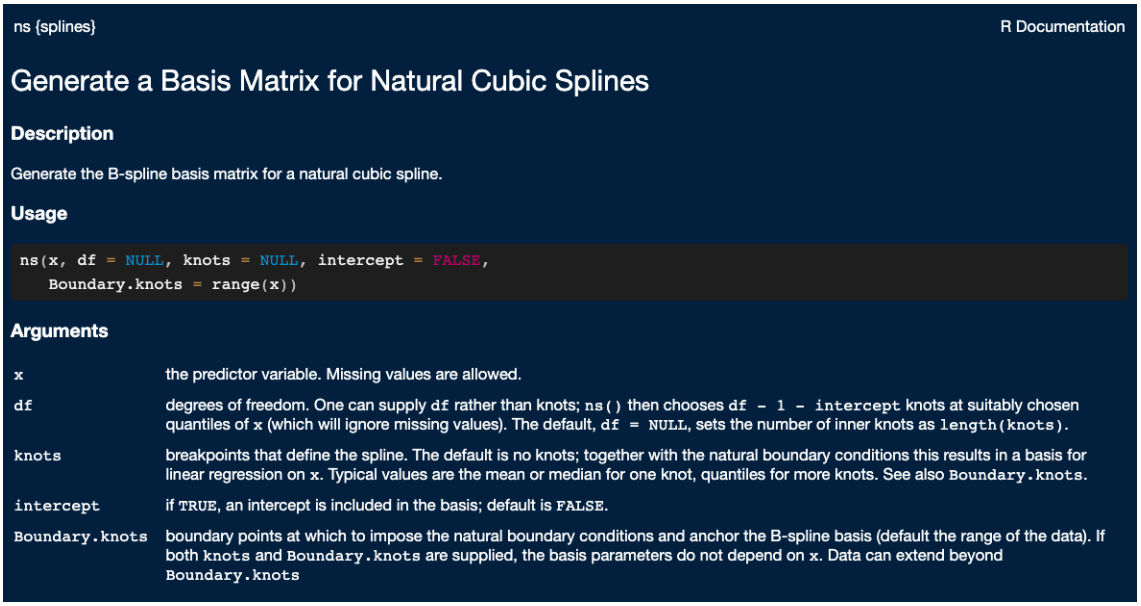
In the 1970s and 80s, statisticians began translating some of these engineering concepts to curve fitting. The basic idea was to create functions of a continuous variable that would yield the appropriate degree of flexibility between that value and the conditional outcome expectation.

When it comes to implementation, there are a great many number of different options one can use to fit splines. Among these include natural cubic splines (the `ns()` option in R), B-splines (the `bs()` option in R), generalized additive models (or GAMs, implemented in the `gam` package or the `mgcv` package in R), penalized smoothing splines (implemented via the `smooth.spline()` function in R), or restricted quadratic splines (Howe et al., 2011).

In R, splines can easily be deployed in a regression context. For example, we can install/load the `splines` package and look at the help page for the `ns()` function:

```
pacman::p_load(splines)

?ns
```



ns {splines} R Documentation

Generate a Basis Matrix for Natural Cubic Splines

Description

Generate the B-spline basis matrix for a natural cubic spline.

Usage

```
ns(x, df = NULL, knots = NULL, intercept = FALSE,
   Boundary.knots = range(x))
```

Arguments

`x`	the predictor variable. Missing values are allowed.
`df`	degrees of freedom. One can supply `df` rather than knots; `ns()` then chooses `df - 1 - intercept` knots at suitably chosen quantiles of `x` (which will ignore missing values). The default, `df = NULL`, sets the number of inner knots as `length(knots)`.
`knots`	breakpoints that define the spline. The default is no knots; together with the natural boundary conditions this results in a basis for linear regression on `x`. Typical values are the mean or median for one knot, quantiles for more knots. See also `Boundary.knots`.
`intercept`	if `TRUE`, an intercept is included in the basis; default is `FALSE`.
`Boundary.knots`	boundary points at which to impose the natural boundary conditions and anchor the B-spline basis (default the range of the data). If both `knots` and `Boundary.knots` are supplied, the basis parameters do not depend on `x`. Data can extend beyond `Boundary.knots`

There are many arguments that we can select from to fit a spline, but we have to pick one of two. The first is the `df` argument. The second is the `knots` argument.

If we work with the `df` argument, we can elect to decrease or increase flexibility by lowering or increasing the degrees of freedom. If we work with the `knots` argument, we can pick the specific points across the distribution of the splined variable. Picking more points increases the flexibility, picking fewer points decreases the flexibility.

Let's apply a spline to our `map` variable. We'll use the `df` argument and we'll pick 4 degrees of freedom:

```
#' splines using the ns() function
model_MD <- glm(wt82_71 ~ qsmk + sex*age + race + income + ns(map, df = 4),
                data = nhefs,
                family = gaussian("identity"))

summary(model_MD)$coefficients
```

```
##                    Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)       6.29160423 2.44954468  2.5684791 1.031291e-02
## qsmk              2.81993961 0.45834677  6.1524151 9.827931e-10
## sex               1.94209894 1.48209535  1.3103738 1.902748e-01
```

```
## age              -0.15504131 0.02330760 -6.6519635 4.072480e-11
## race             -0.25915103 0.60897312 -0.4255541 6.704953e-01
## income            0.04586951 0.07779011  0.5896573 5.555113e-01
## ns(map, df = 4)1  2.57736544 1.56462276  1.6472759 9.971579e-02
## ns(map, df = 4)2  2.58271283 1.48604525  1.7379772 8.242494e-02
## ns(map, df = 4)3  1.69399520 3.81428137  0.4441191 6.570221e-01
## ns(map, df = 4)4  3.01573103 3.31088944  0.9108522 3.625231e-01
## sex:age          -0.04512872 0.03266318 -1.3816387 1.672933e-01
```

2.4 Extracting Point and Variance Estimates

Using the tidy function, we can save the estimates and standard errors that we want to an object in R.

```
mean_difference1 <- tidy(model_MD)[2,]

mean_difference1
```

```
## # A tibble: 1 x 5
##   term  estimate std.error statistic  p.value
##   <chr>    <dbl>     <dbl>     <dbl>    <dbl>
## 1 qsmk      2.82     0.458      6.15 9.83e-10
```

If desired, we can construct confidence intervals using the Wald (or normal interval) equation as follows:

```
mean_difference1_LCL <- mean_difference1$estimate - 1.96*mean_difference1$std.error
mean_difference1_UCL <- mean_difference1$estimate + 1.96*mean_difference1$std.error

round(c(mean_difference1_LCL, mean_difference1_UCL), 2)
```

```
## [1] 1.92 3.72
```

3 Conditionally Adjusted Regression Model for the Odds Ratio

3.1 Logistic Regression with Interactions and Splines

Next, lets estimate the effect of quitting smoking on a dichotomized weight gain variable using a conditionally adjusted logistic regression model. We'll include an interaction between sex and age, and we'll include splines for the mean arterial pressure variable we constructed:

```
nhefs <- nhefs %>%
  mutate(wt_delta = as.numeric(wt82_71 > median(wt82_71)))

#' This model can be used to quantify a conditionally adjusted OR
#' logit model for the OR with correct standard error
#' including an interaction and spline
model_OR <- glm(wt_delta ~ qsmk + sex*age + race + income + ns(map, df=4),
                data = nhefs,
                family = binomial("logit"))

#' summary() again
summary(model_OR)
```

```
##
## Call:
## glm(formula = wt_delta ~ qsmk + sex * age + race + income + ns(map,
##     df = 4), family = binomial("logit"), data = nhefs)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       1.000148   0.682629   1.465    0.143
## qsmk              0.610771   0.129280   4.724 2.31e-06 ***
## sex               0.388744   0.420566   0.924    0.355
## age              -0.038192   0.006655  -5.739 9.55e-09 ***
## race             -0.190564   0.169507  -1.124    0.261
## income            0.022717   0.021693   1.047    0.295
## ns(map, df = 4)1  0.230793   0.436488   0.529    0.597
```

```
## ns(map, df = 4)2  0.631591   0.411583   1.535    0.125
## ns(map, df = 4)3 -0.151820   1.060306  -0.143    0.886
## ns(map, df = 4)4  0.519238   0.905165   0.574    0.566
## sex:age          -0.006577   0.009337  -0.704    0.481
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2046.2  on 1475  degrees of freedom
## Residual deviance: 1936.5  on 1465  degrees of freedom
## AIC: 1958.5
##
## Number of Fisher Scoring iterations: 4
```

```
#' tidy() again
tidy(model_OR)
```

```
## # A tibble: 11 x 5
##    term              estimate std.error statistic       p.value
##    <chr>                <dbl>     <dbl>     <dbl>         <dbl>
##  1 (Intercept)       1.00      0.683       1.47  0.143
##  2 qsmk              0.611     0.129       4.72  0.00000231
##  3 sex               0.389     0.421       0.924 0.355
##  4 age              -0.0382    0.00666    -5.74  0.00000000955
##  5 race             -0.191     0.170      -1.12  0.261
##  6 income            0.0227    0.0217      1.05  0.295
##  7 ns(map, df = 4)1  0.231     0.436       0.529 0.597
##  8 ns(map, df = 4)2  0.632     0.412       1.53  0.125
##  9 ns(map, df = 4)3 -0.152     1.06       -0.143 0.886
## 10 ns(map, df = 4)4  0.519     0.905       0.574 0.566
## 11 sex:age          -0.00658   0.00934    -0.704 0.481
```

To get the OR from the regression model, we have to exponentiate the coefficient from the model output. We can also construct confidence intervals in the same way:

```
qsmk_log_OR <- tidy(model_OR)[2,]

qsmk_OR <- exp(qsmk_log_OR[1,2])

qsmk_OR
```

```
## # A tibble: 1 x 1
##   estimate
##      <dbl>
## 1     1.84
```

```
qsmk_OR_LCL <- exp(qsmk_log_OR$estimate - 1.96*qsmk_log_OR$std.error)
qsmk_OR_UCL <- exp(qsmk_log_OR$estimate + 1.96*qsmk_log_OR$std.error)

round(c(qsmk_OR$estimate, qsmk_OR_LCL, qsmk_OR_UCL),2)
```

```
## [1] 1.84 1.43 2.37
```

If we were interested in estimating conditionally adjusted risk differences or risk ratios for the effect of quitting smoking on greater than median weight change, we could use a similar approach with the identity link function, ordinary least squares, or Poisson regression (Zou, 2004, Naimi and Whitcomb (2020)). However, we have to ensure that the variance estimators are appropriate.

Unfortunately, it's not always clear **how** to get correct variances that are correct when using identity link functions, ordinary least squares, or the Poisson distribution. The following Table provides a brief heuristic outline of how one might do this correctly.

Table 1: Methods to use for quantifying conditionally adjusted odds ratios, risk ratios, and risk differences.

Odds Ratio	Risk Ratio	Risk Difference
GLM Family = Binomial	GLM Family = Binomial	GLM Family = Binomial
GLM Link = Logistic	GLM Link = Log	GLM Link = Identity

Odds Ratio	Risk Ratio	Risk Difference
Standard Errors = Model Based	Standard Errors = Model Based	Standard Errors = Model Based
	GLM Family = Poisson	GLM Family = Gaussian
	GLM Link = Log	GLM Link = Identity
	Standard Errors = Sandwich	Standard Errors = Sandwich
		Least Squares Regression
		Standard Errors = Sandwich

The procedures above constitute general procedures in which we can use the `glm` function to estimate associations.

4 Marginally Adjusted Regression Model

Another approach to obtaining mean differences, risk differences, and risk ratios from GLMs is to use marginal standardization (Naimi et al., 2017). This process can be implemented by fitting a single model, regressing the outcome against the exposure and all confounder variables. But instead of reading the coefficients the model, one can obtain odds ratios, risk ratios, or risk differences by using this model to generate predicted risks for each individual under "exposed" and "unexposed" scenarios in the dataset. To obtain standard errors, the entire procedure must be bootstrapped.

Here is some code to implement this marginal standardization in the NHEFS data for the association between quitting smoking and weight change:

```
#' Regress the outcome against the confounders with interaction
model_MD <- glm(wt82_71 ~ qsmk + sex + age +
    race + income + map, data = nhefs, family = gaussian("identity"))
##' Generate predictions for everyone in the sample to obtain
##' unexposed (mu0 predictions) and exposed (mu1 predictions) risks.
mu1 <- predict(model_MD, newdata = transform(nhefs,
    qsmk = 1), type = "response")
mu0 <- predict(model_MD, newdata = transform(nhefs,
    qsmk = 0), type = "response")
```

```r
#' Marginally adjusted mean difference
marg_stand_MD <- mean(mu1) - mean(mu0)

#' Using the bootstrap to obtain confidence intervals for the marginally adjusted
#' risk ratio and risk difference.
bootfunc <- function(data, index) {
    boot_dat <- data[index, ]
    model_MD_ <- glm(wt82_71 ~ qsmk + sex +
        age + race + income + map, data = boot_dat,
        family = gaussian("identity"))
    mu1_ <- predict(model_MD_, newdata = transform(boot_dat,
        qsmk = 1), type = "response")
    mu0_ <- predict(model_MD_, newdata = transform(boot_dat,
        qsmk = 0), type = "response")

    #' Marginally adjusted mean difference
    res <- mean(mu1_) - mean(mu0_)
    return(res)
}

#' Run the boot function. Set a seed to obtain reproducibility
set.seed(123)
boot_res <- boot(nhefs, bootfunc, R = 2000)

boot_MD <- boot.ci(boot_res)

marg_stand_MD
```

```
## [1] 2.832019
```

```r
boot_MD
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
```

```
## CALL :
## boot.ci(boot.out = boot_res)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 1.835,  3.812 )   ( 1.866,  3.793 )
##
## Level     Percentile            BCa
## 95%   ( 1.871,  3.798 )   ( 1.862,  3.794 )
## Calculations and Intervals on Original Scale
```

We can do the same thing to estimate the association between quitting smoking and death:

```
#' Regress the outcome against the confounders with interaction
model_OR <- glm(death ~ qsmk + sex + age +
    race + income + map, data = nhefs, family = binomial("logit"))
##' Generate predictions for everyone in the sample to obtain
##' unexposed (mu0 predictions) and exposed (mu1 predictions) risks.
mu1 <- predict(model_OR, newdata = transform(nhefs,
    qsmk = 1), type = "response")
mu0 <- predict(model_OR, newdata = transform(nhefs,
    qsmk = 0), type = "response")

#' Marginally adjusted odds ratio
marg_stand_OR <- (mean(mu1)/mean(1 - mu1))/(mean(mu0)/mean(1 -
    mu0))
#' Marginally adjusted risk ratio
marg_stand_RR <- mean(mu1)/mean(mu0)
#' Marginally adjusted risk difference
marg_stand_RD <- mean(mu1) - mean(mu0)

#' Using the bootstrap to obtain confidence intervals for the marginally adjusted
#' risk ratio and risk difference.
bootfunc <- function(data, index) {
    boot_dat <- data[index, ]
```

```
    model_OR_ <- glm(death ~ qsmk + sex +
        age + race + income + map, data = boot_dat,
        family = binomial("logit"))
    mu1 <- predict(model_OR_, newdata = transform(boot_dat,
        qsmk = 1), type = "response")
    mu0 <- predict(model_OR_, newdata = transform(boot_dat,
        qsmk = 0), type = "response")

    marg_stand_OR_ <- (mean(mu1)/mean(1 -
        mu1))/(mean(mu0)/mean(1 - mu0))
    marg_stand_RR_ <- mean(mu1)/mean(mu0)
    marg_stand_RD_ <- mean(mu1) - mean(mu0)
    res <- c(marg_stand_RD_, marg_stand_RR_,
        marg_stand_OR_)
    return(res)
}

#' Run the boot function. Set a seed to obtain reproducibility
set.seed(123)
boot_res <- boot(nhefs, bootfunc, R = 2000)

boot_RD <- boot.ci(boot_res, index = 1)
boot_RR <- boot.ci(boot_res, index = 2)
boot_OR <- boot.ci(boot_res, index = 3)

marg_stand_OR
```

```
## [1] 1.001926
```

```
marg_stand_RR
```

```
## [1] 1.001576
```

```
marg_stand_RD
```

```
## [1] 0.0002860808
```

```
boot_RD
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_res, index = 1)
##
## Intervals :
## Level      Normal              Basic
## 95%   (-0.0379,  0.0389 )   (-0.0400,  0.0378 )
##
## Level     Percentile            BCa
## 95%   (-0.0373,  0.0406 )   (-0.0354,  0.0424 )
## Calculations and Intervals on Original Scale
```

```
boot_RR
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_res, index = 2)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.786,  1.213 )   ( 0.765,  1.200 )
##
## Level     Percentile            BCa
## 95%   ( 0.803,  1.238 )   ( 0.814,  1.249 )
## Calculations and Intervals on Original Scale
```

```
boot_OR
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 2000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_res, index = 3)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.735,  1.259 )   ( 0.705,  1.238 )
##
## Level     Percentile            BCa
## 95%   ( 0.766,  1.299 )   ( 0.780,  1.314 )
## Calculations and Intervals on Original Scale
```

References

CJ Howe, SR Cole, DJ Westreich, S Greenland, S Napravnik, and JJ Eron. Splines for trend analysis and continuous confounder control. *Epidemiol*, 22(6):874–5, 2011.

Ashley I Naimi and Brian W Whitcomb. Estimating risk ratios and risk differences using regression. *American Journal of Epidemiology*, 189(6):508–510, 2020.

Ashley I Naimi, Stephen R Cole, and Edward H Kennedy. An Introduction to G Methods. *Int J Epidemiol*, 46(2):756–62, 2017.

Guangyong Zou. A modified poisson regression approach to prospective studies with binary data. *Am J Epidemiol*, 159(7):702–706, Apr 2004.