**Instructions**

There is another approach for assessing the validity of ANOVAs. We usually interpret the F-test by looking at the significance level of the test. This significance level is determined by the p-value, which reflects the probability of finding a less likely result under the null hypothesis - i.e. when there is no effect. As opposed to the F-test itself, its corresponding p-value behaves consistently under the null hypothesis. More specifically, for any ANOVA based on any given data set under the null hypothesis, the corresponding p-value has a uniform distribution, an expected value of .50 and requires the p-value to be confidence valid (a p-value of .05 should only occur 5 percent of the time).

What I need to understand:

**How to examine the following about the p-value:**

- Has an expected value of .50 (why .50 to look at the bias? Which table of the results do I have to use to evaluate the bias, see output )

- Should be confidence valid (Neyman, 1934) in the sense that a p-value of .05 should only occur 5 percent of the time. (how to check this using my results)

**Interpretation of the results:**

**- Expected value of the p-values: text with expected values and table + interpretation**

**- Proportion E(p<=.05): text + table**

**- Proportion correct** (i.e. how many p-values less than .05 are the same as the population)

- what can I say about the reliability? And the power?

- Why use an intercept only model in the code (model with only the intercept)

**Those results can be obtained by using the R script. I added some in the info document.**

R version 4.0.1 (2020-06-06) -- "See Things Now"

Copyright (C) 2020 The R Foundation for Statistical Computing

Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.

You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.

Type 'contributors()' for more information and

'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or

'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

> library(mice)

Attaching package: ‘mice’

The following objects are masked from ‘package:base’:

cbind, rbind

> library(miceadds)

\* miceadds 3.9-14 (2020-05-09 11:27:27)

> library(magrittr)

> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

filter, lag

The following objects are masked from ‘package:base’:

intersect, setdiff, setequal, union

> library(purrr)

Attaching package: ‘purrr’

The following object is masked from ‘package:magrittr’:

set\_names

> library(mvtnorm)

> library(ggplot2)

Need help getting started? Try the R Graphics Cookbok:

https://r-graphics.org

> #install.packages("ggplot2")

> set.seed(123)

>

> # Simulation parameters

> nsim = 1000

> rho = 0 # correlation set to zero to conform to null hypothesis (no effect)

>

> # Generate data function

> make.data <- function(n, correlation){

+ data <- rmvnorm(n = n, mean = c(0, 0),

+ sigma = matrix(c(1, correlation, correlation, 1),

+ nrow = 2, ncol = 2))

+ colnames(data) <- c("y", "x")

+ data %>% as\_tibble() %>% return()

+ }

>

> # simulation function

> simulate <- function(n, rho, prop.mis = .75){

+ # sample data from multivariate normal distribution

+ data <- make.data(n = n, correlation = rho)

+ # ampute data

+ missing <- ampute(data,

+ patterns = matrix(c(0, 1, 1, 0), ncol = 2, nrow = 2, byrow = TRUE),

+ prop = prop.mis,

+ mech = "MCAR")

+ # impute data

+ imp <- mice(missing$amp, method = "norm", m = 10, maxit = 10, print = FALSE)

+ return(list(data = data,

+ miss = missing,

+ imp = imp))

+ }

>

> # run simulation

> SIM01 <- replicate(nsim, simulate(n = 1000, rho = rho, prop.mis = .01), simplify = FALSE)

> SIM25 <- replicate(nsim, simulate(n = 1000, rho = rho, prop.mis = .25), simplify = FALSE)

> SIM50 <- replicate(nsim, simulate(n = 1000, rho = rho, prop.mis = .50), simplify = FALSE)

> SIM75 <- replicate(nsim, simulate(n = 1000, rho = rho, prop.mis = .75), simplify = FALSE)

>

> #evaluate

> # Read in the simulation data

>

> # evaluation function

> evaluate <- function(sim){

+ # calculate true data statistics

+ fit <- sim$data %$% lm(y ~ x)

+ anova <- fit %>% anova

+ # calculate observed data statistics

+ fit.mis <- sim$miss$amp %$% lm(y ~ x)

+ anova.mis <- fit.mis %>% anova

+ # calculate imputed data statistics

+ fit.imp <- with(data = sim$imp, expr = lm(y ~ x))

+ fit.imp.empty <- with(data = sim$imp, expr = lm(y ~ 1))

+ fit.imp.pool <- pool(fit.imp)

+ fit.anova.D1 <- D1(fit.imp, fit.imp.empty)

+ fit.anova.D2 <- D2(fit.imp, fit.imp.empty) #cf. micombine.F from miceadds

+ fit.anova.D3 <- D3(fit.imp, fit.imp.empty)

+ F.vector <- unlist(with(sim$imp, anova(lm(y ~ x))$'F value'[1])$analyses)

+ micomb <- micombine.F(F.vector, anova$Df[[1]], display = FALSE)

+ avg.F.imp <- mean(F.vector)

+ avg.p.imp <- mean(unlist(with(sim$imp, anova(lm(y ~ x))$'Pr(>F)'[1])$analyses))

+ # prepare output for return

+ return(list(truefit = fit,

+ trueanova = anova,

+ missfit = fit.mis,

+ missanova = anova.mis,

+ fit = fit.imp,

+ emptyfit = fit.imp.empty,

+ pool = fit.imp.pool,

+ D1 = fit.anova.D1,

+ D2 = fit.anova.D2,

+ D3 = fit.anova.D3,

+ micomb = round(micomb, 4),

+ Fvector = F.vector,

+ Fbar = avg.F.imp,

+ pbar = avg.p.imp))

+ }

>

> EVAL01 <- map(SIM01, evaluate)

> EVAL25 <- map(SIM25, evaluate)

> EVAL50 <- map(SIM50, evaluate)

> EVAL75 <- map(SIM75, evaluate)

>

> # Grab Anova's

> grab.F <- function(x){

+ data.frame(true = x$trueanova$`F value`[1],

+ mis = x$missanova$`F value`[1],

+ D1 = x$D1$result[1],

+ D2 = x$D2$result[1],

+ D3 = x$D3$result[1],

+ micombine = x$micomb[1],

+ Fbar = x$Fbar)

+ }

> F.out01 <- mapply(grab.F, EVAL01, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> F.out25 <- mapply(grab.F, EVAL25, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> F.out50 <- mapply(grab.F, EVAL50, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> F.out75 <- mapply(grab.F, EVAL75, SIMPLIFY = FALSE) %>% do.call(rbind, .)

>

> # Grab p-values

> grab.p <- function(x){

+ data.frame(true = x$trueanova$`Pr(>F)`[1],

+ mis = x$missanova$`Pr(>F)`[1],

+ D1 = x$D1$result[4],

+ D2 = x$D2$result[4],

+ D3 = x$D3$result[4],

+ micombine = x$micomb[2],

+ pbar = x$pbar)

+ }

> p.out01 <- mapply(grab.p, EVAL01, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> p.out25 <- mapply(grab.p, EVAL25, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> p.out50 <- mapply(grab.p, EVAL50, SIMPLIFY = FALSE) %>% do.call(rbind, .)

> p.out75 <- mapply(grab.p, EVAL75, SIMPLIFY = FALSE) %>% do.call(rbind, .)

>

> # Save the evaluations and the resulting p-value and F-value objects

> #save(list = c("EVAL01", "EVAL25", "EVAL50", "EVAL75"),

> #file = "Evaluations\_objects.RData")

> #save(list = c("F.out01", "F.out25", "F.out50", "F.out75",

> #"p.out01", "p.out25", "p.out50", "p.out75"),

> #file = "Evaluations\_processed.RData")

>

>

> # Test whether the p-values follow a uniform distribution

> apply(p.out01, 2, function(x) ks.test(x, "punif", 0, 1))

$true

One-sample Kolmogorov-Smirnov test

data: x

D = 0.016054, p-value = 0.9588

alternative hypothesis: two-sided

$mis

One-sample Kolmogorov-Smirnov test

data: x

D = 0.015352, p-value = 0.9725

alternative hypothesis: two-sided

$D1

One-sample Kolmogorov-Smirnov test

data: x

D = 0.01627, p-value = 0.9539

alternative hypothesis: two-sided

$D2

One-sample Kolmogorov-Smirnov test

data: x

D = 0.027795, p-value = 0.4224

alternative hypothesis: two-sided

$D3

One-sample Kolmogorov-Smirnov test

data: x

D = 0.015988, p-value = 0.9603

alternative hypothesis: two-sided

$micombine

One-sample Kolmogorov-Smirnov test

data: x

D = 0.0278, p-value = 0.4222

alternative hypothesis: two-sided

$pbar

One-sample Kolmogorov-Smirnov test

data: x

D = 0.036678, p-value = 0.1356

alternative hypothesis: two-sided

Warning message:

In ks.test(x, "punif", 0, 1) :

ties should not be present for the Kolmogorov-Smirnov test

> apply(p.out25, 2, function(x) ks.test(x, "punif", 0, 1))

$true

One-sample Kolmogorov-Smirnov test

data: x

D = 0.022583, p-value = 0.6876

alternative hypothesis: two-sided

$mis

One-sample Kolmogorov-Smirnov test

data: x

D = 0.026881, p-value = 0.4652

alternative hypothesis: two-sided

$D1

One-sample Kolmogorov-Smirnov test

data: x

D = 0.037645, p-value = 0.1175

alternative hypothesis: two-sided

$D2

One-sample Kolmogorov-Smirnov test

data: x

D = 0.1768, p-value < 2.2e-16

alternative hypothesis: two-sided

$D3

One-sample Kolmogorov-Smirnov test

data: x

D = 0.037641, p-value = 0.1176

alternative hypothesis: two-sided

$micombine

One-sample Kolmogorov-Smirnov test

data: x

D = 0.1768, p-value < 2.2e-16

alternative hypothesis: two-sided

$pbar

One-sample Kolmogorov-Smirnov test

data: x

D = 0.2282, p-value < 2.2e-16

alternative hypothesis: two-sided

Warning message:

In ks.test(x, "punif", 0, 1) :

ties should not be present for the Kolmogorov-Smirnov test

> apply(p.out50, 2, function(x) ks.test(x, "punif", 0, 1))

$true

One-sample Kolmogorov-Smirnov test

data: x

D = 0.019629, p-value = 0.8357

alternative hypothesis: two-sided

$mis

One-sample Kolmogorov-Smirnov test

data: x

D = 0.026267, p-value = 0.4952

alternative hypothesis: two-sided

$D1

One-sample Kolmogorov-Smirnov test

data: x

D = 0.035301, p-value = 0.1653

alternative hypothesis: two-sided

$D2

One-sample Kolmogorov-Smirnov test

data: x

D = 0.28603, p-value < 2.2e-16

alternative hypothesis: two-sided

$D3

One-sample Kolmogorov-Smirnov test

data: x

D = 0.035405, p-value = 0.1629

alternative hypothesis: two-sided

$micombine

One-sample Kolmogorov-Smirnov test

data: x

D = 0.286, p-value < 2.2e-16

alternative hypothesis: two-sided

$pbar

One-sample Kolmogorov-Smirnov test

data: x

D = 0.36058, p-value < 2.2e-16

alternative hypothesis: two-sided

Warning message:

In ks.test(x, "punif", 0, 1) :

ties should not be present for the Kolmogorov-Smirnov test

> apply(p.out75, 2, function(x) ks.test(x, "punif", 0, 1))

$true

One-sample Kolmogorov-Smirnov test

data: x

D = 0.028412, p-value = 0.3948

alternative hypothesis: two-sided

$mis

One-sample Kolmogorov-Smirnov test

data: x

D = 0.024604, p-value = 0.5802

alternative hypothesis: two-sided

$D1

One-sample Kolmogorov-Smirnov test

data: x

D = 0.024617, p-value = 0.5796

alternative hypothesis: two-sided

$D2

One-sample Kolmogorov-Smirnov test

data: x

D = 0.3918, p-value < 2.2e-16

alternative hypothesis: two-sided

$D3

One-sample Kolmogorov-Smirnov test

data: x

D = 0.02465, p-value = 0.5778

alternative hypothesis: two-sided

$micombine

One-sample Kolmogorov-Smirnov test

data: x

D = 0.3918, p-value < 2.2e-16

alternative hypothesis: two-sided

$pbar

One-sample Kolmogorov-Smirnov test

data: x

D = 0.51477, p-value < 2.2e-16

alternative hypothesis: two-sided

Warning messages:

1: In ks.test(x, "punif", 0, 1) :

ties should not be present for the Kolmogorov-Smirnov test

2: In ks.test(x, "punif", 0, 1) :

ties should not be present for the Kolmogorov-Smirnov test

>

> # Test how many p=values <.05 (we expect .05)

> # apply(p.out25, 2, function(x) mean(x < .05))

>

> data.frame(mis01 = apply(p.out01, 2, function(x) mean(x < .05)),

+ mis25 = apply(p.out25, 2, function(x) mean(x < .05)),

+ mis50 = apply(p.out50, 2, function(x) mean(x < .05)),

+ mis75 = apply(p.out75, 2, function(x) mean(x < .05)))

mis01 mis25 mis50 mis75

true 0.050 0.058 0.054 0.047

mis 0.050 0.045 0.037 0.054

D1 0.051 0.043 0.042 0.048

D2 0.051 0.043 0.040 0.046

D3 0.051 0.043 0.045 0.048

micombine 0.051 0.043 0.040 0.046

pbar 0.051 0.053 0.073 0.112

>

> # What is the mean p-value

> # apply(p.out, 2, function(x) mean(x))

>

> data.frame(mis01 = apply(p.out01, 2, function(x) mean(x)),

+ mis25 = apply(p.out25, 2, function(x) mean(x)),

+ mis50 = apply(p.out50, 2, function(x) mean(x)),

+ mis75 = apply(p.out75, 2, function(x) mean(x)))

mis01 mis25 mis50 mis75

true 0.4982534 0.5034517 0.5036847 0.5135671

mis 0.4981362 0.5126605 0.5076661 0.4903709

D1 0.4993402 0.5171933 0.5164205 0.4932186

D2 0.4997744 0.4691195 0.4057441 0.3144516

D3 0.4989442 0.5169476 0.5161389 0.4930915

micombine 0.4997742 0.4691190 0.4057441 0.3144517

pbar 0.4962151 0.4295647 0.3431967 0.2251221

>

> # Bias in p-value with respect to the population p=value

> # population = .5 under null hypothesis

> # apply(p.out, 2, function(x) mean(x - .5))

>

> data.frame(mis01 = apply(p.out01, 2, function(x) mean(x - .5)),

+ mis25 = apply(p.out25, 2, function(x) mean(x - .5)),

+ mis50 = apply(p.out50, 2, function(x) mean(x - .5)),

+ mis75 = apply(p.out75, 2, function(x) mean(x - .5)))

mis01 mis25 mis50 mis75

true -0.0017465770 0.00345171 0.003684722 0.013567127

mis -0.0018638391 0.01266052 0.007666078 -0.009629126

D1 -0.0006597744 0.01719326 0.016420507 -0.006781448

D2 -0.0002255575 -0.03088055 -0.094255891 -0.185548421

D3 -0.0010558269 0.01694759 0.016138877 -0.006908521

micombine -0.0002258000 -0.03088100 -0.094255900 -0.185548300

pbar -0.0037848667 -0.07043531 -0.156803261 -0.274877939

>

> # Bias in p-value with respect to the sampled complete p=value

> # population = .5 under null hypothesis

> # truth <- p.out$true

> # apply(p.out, 2, function(x) mean(x - truth))

>

> truth01 <- p.out01$true

> truth25 <- p.out25$true

> truth50 <- p.out50$true

> truth75 <- p.out75$true

> data.frame(mis01 = apply(p.out01, 2, function(x) mean(x - truth01)),

+ mis25 = apply(p.out25, 2, function(x) mean(x - truth25)),

+ mis50 = apply(p.out50, 2, function(x) mean(x - truth50)),

+ mis75 = apply(p.out75, 2, function(x) mean(x - truth75)))

mis01 mis25 mis50 mis75

true 0.0000000000 0.000000000 0.000000000 0.00000000

mis -0.0001172621 0.009208812 0.003981356 -0.02319625

D1 0.0010868026 0.013741546 0.012735786 -0.02034858

D2 0.0015210195 -0.034332258 -0.097940613 -0.19911555

D3 0.0006907501 0.013495876 0.012454156 -0.02047565

micombine 0.0015207770 -0.034332710 -0.097940622 -0.19911543

pbar -0.0020382897 -0.073887024 -0.160487983 -0.28844507

>

> # Bias in p-value with respect to the sampled incomplete p=value

> # population = .5 under null hypothesis

> # mis <- p.out$mis

> # apply(p.out, 2, function(x) mean(x - mis))

>

> mis01 <- p.out01$mis

> mis25 <- p.out25$mis

> mis50 <- p.out50$mis

> mis75 <- p.out75$mis

> data.frame(mis01 = apply(p.out01, 2, function(x) mean(x - mis01)),

+ mis25 = apply(p.out25, 2, function(x) mean(x - mis25)),

+ mis50 = apply(p.out50, 2, function(x) mean(x - mis50)),

+ mis75 = apply(p.out75, 2, function(x) mean(x - mis75)))

mis01 mis25 mis50 mis75

true 0.0001172621 -0.009208812 -0.003981356 0.023196253

mis 0.0000000000 0.000000000 0.000000000 0.000000000

D1 0.0012040647 0.004532734 0.008754430 0.002847677

D2 0.0016382816 -0.043541071 -0.101921969 -0.175919295

D3 0.0008080122 0.004287064 0.008472800 0.002720605

micombine 0.0016380391 -0.043541522 -0.101921978 -0.175919174

pbar -0.0019210276 -0.083095837 -0.164469339 -0.265248813

>

> # Correlations with truth

> cor(p.out01)[1, ]

true mis D1 D2 D3 micombine pbar

1.0000000 0.9797005 0.9781189 0.9784891 0.9781087 0.9784894 0.9785910

> cor(p.out25)[1, ]

true mis D1 D2 D3 micombine pbar

1.0000000 0.6344155 0.6073024 0.6290787 0.6069431 0.6290811 0.6279764

> cor(p.out50)[1, ]

true mis D1 D2 D3 micombine pbar

1.0000000 0.3663182 0.3545228 0.3434731 0.3537066 0.3434758 0.3472681

> cor(p.out75)[1, ]

true mis D1 D2 D3 micombine pbar

1.0000000 0.1822120 0.1407165 0.1623875 0.1409298 0.1623900 0.1677232

>

> # proportions correct

> sign01 <- p.out01 < .05

> sign25 <- p.out25 < .05

> sign50 <- p.out50 < .05

> sign75 <- p.out75 < .05

>

> p.correct01 <- apply(sign01, 2, function(x) mean(x == sign01[, 1]))

> p.correct25 <- apply(sign25, 2, function(x) mean(x == sign25[, 1]))

> p.correct50 <- apply(sign50, 2, function(x) mean(x == sign50[, 1]))

> p.correct75 <- apply(sign75, 2, function(x) mean(x == sign75[, 1]))

>

> #Plots

> library(ggplot2)

> library(gridExtra)

Error in library(gridExtra) : there is no package called ‘gridExtra’

>

> # plot function

> qqplot <- function(x, title = "ECDF and theoretical CDF"){

+ ggplot(aes(x), data = data.frame(x)) +

+ stat\_function(fun=punif,

+ args=list(0, 1),

+ col = "orange",

+ lwd = 1.1) +

+ stat\_ecdf() +

+ labs(title = title) +

+ xlab("Empirical CDF") +

+ ylab("CDF") +

+ xlim(0, 1)

+ }

>

> # 01% missingness

> qqplot(p.out01$true)

> qqplot(p.out01$mis)

> qqplot(p.out01$D1)

> qqplot(p.out01$D2)

> qqplot(p.out01$D3)

> qqplot(p.out01$micombine)

> qqplot(p.out01$pbar)

>

> # 25% missingness

> qqplot(p.out25$true)

> qqplot(p.out25$mis)

> qqplot(p.out25$D1)

> qqplot(p.out25$D2)

> qqplot(p.out25$D3)

> qqplot(p.out25$micombine)

> qqplot(p.out25$pbar)

>

> # 50% missingness

> qqplot(p.out50$true)

> qqplot(p.out50$mis)

> qqplot(p.out50$D1)

> qqplot(p.out50$D2)

> qqplot(p.out50$D3)

> qqplot(p.out50$micombine)

> qqplot(p.out50$pbar)

>

> # 75% missingness

> qqplot(p.out75$true)

> qqplot(p.out75$mis)

> qqplot(p.out75$D1)

> qqplot(p.out75$D2)

> qqplot(p.out75$D3)

> qqplot(p.out75$micombine)

> qqplot(p.out75$pbar)

>

> #side by side

> p1 <- qqplot(p.out75$D1, title = "D1")

> p2 <- qqplot(p.out75$D2, title = "D2")

> p3 <- qqplot(p.out75$D3, title = "D3")

> grid.arrange(p1, p2, p3,

+ nrow = 1, respect=TRUE,

+ top = "ECDF and theoretical CDF")

Error in grid.arrange(p1, p2, p3, nrow = 1, respect = TRUE, top = "ECDF and theoretical CDF") :

could not find function "grid.arrange"

Error in UseMethod("depth") :

no applicable method for 'depth' applied to an object of class "NULL"

> >