Simulations

Lauren Quesada

10/22/2021

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(broom)
LFunky<- function(beta0,beta1,beta2,n_obs,n_data,n_shuffle) {

  set.seed(4747)
  sig<- 0.05


  #If there is no intercept in the fitted regression, d.o.f. = (n-1)
  #otherwise, df=n-k-1; k=the number of variables
  dof <- n_obs-1

  #addp_val<-c()

  #intp_val<-c()

  simpp_val<-c()

  simp_t<-c()

  perm_pval<-c()


  for(i in 1:n_data) {               # Start of the Monte Carlo loop

  x1<- rnorm(n_obs)     # Artificial x series, created just once

  x2<- x1+ rnorm(n_obs)     ## Did we figure out why these need to be defined in the loop?
```

```
  y<- beta0 + beta1*x1 + beta2*x2 + rnorm(n_obs)   # The DGP includes an intercept

  simpfit<- lm(y ~ x1  -1)

  #observed p.value for each dataset (n_data)
  simpp_val<- c(simpp_val,
                simpfit %>% tidy() %>% filter(term =="x1") %>% select(p.value) %>% pull())

  #observed slope for each dataset (n_data)
  simp_t<- c(simp_t,
                simpfit %>% tidy() %>% filter(term =="x1") %>% select(statistic) %>% pull())

  #obs value for most recent dataset
  obsval<- simpfit %>% tidy() %>% filter(term =="x1") %>% select(statistic) %>% pull()

  #reassign vector so it doesn't add onto the concatenation each i in 1:n_data loop
  simpp_valperm<-c()

    for(j in 1: n_shuffle) {
  # A mis-specified model is estimated (unless Beta0 = 0)
    #addfit<- lm(y ~ x1+x2  -1)

    #intfit<- lm(y ~ x1*x2  -1)

    simpfitperm<- lm(y ~ sample(x1)  -1)

    #addp_val<- c(addp_val, unname(coef(summary(addfit))[, "Pr(>|t|)"]))

    #intp_val<- c(intp_val, unname(coef(summary(intfit))[, "Pr(>|t|)"]))


    simpp_valperm<- c(simpp_valperm,
                      simpfitperm %>% tidy() %>% filter(term == "sample(x1)") %>% select(statistic) %>%

    }

  #p-value for each of my 200 datasets as compared to my permuted distribution of p-values
  perm_pval<-c(perm_pval,mean(simpp_valperm>obsval))

  }  #End of the Monte Carlo Loop

#coefficients, t-test p.values, permutation p.values
data.frame(simp_t, simpp_val, perm_pval)




}

#hist(addp_val, main="Distribution of  *ADDITIVE* p-Values", xlab="p-Value", freq=FALSE, border="black"

#hist(intp_val, main="Distribution of *INTERACTIVE* p-Values", xlab="p-Value", freq=FALSE, border="blac
```

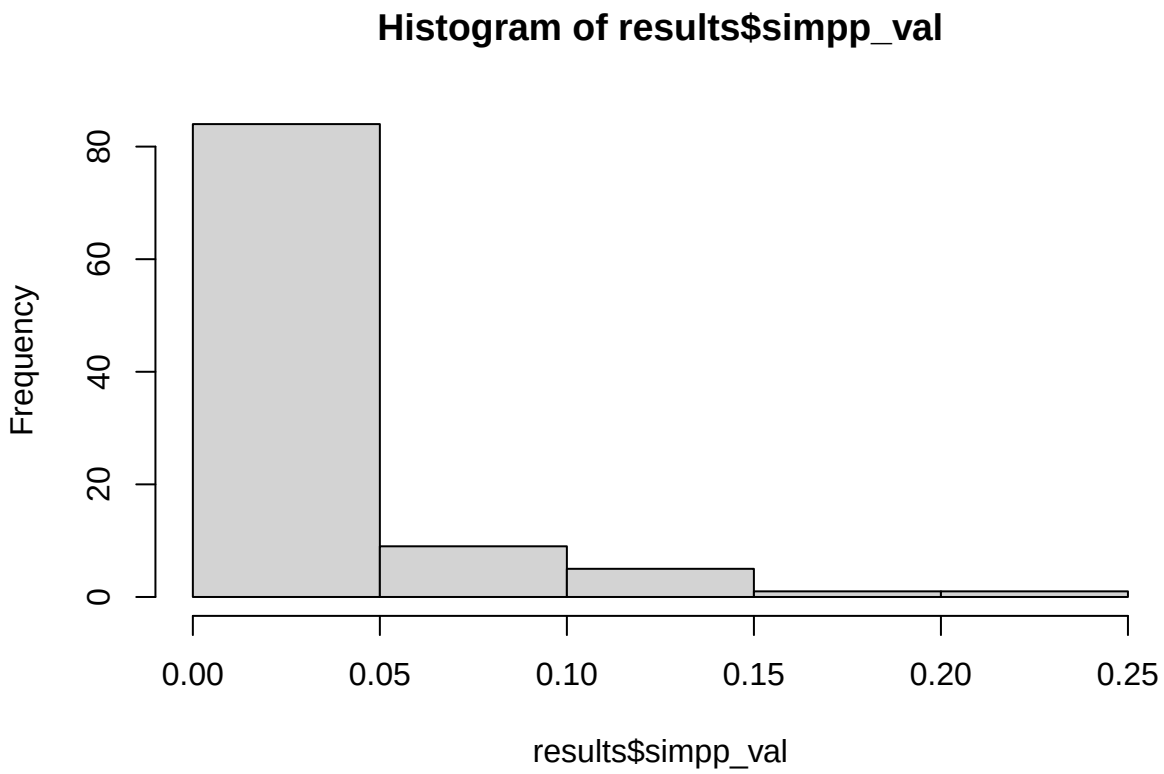
```
#hist(simpp_val, main="Distribution of *SIMPLE* p-Values", xlab="p-Value", freq=FALSE, border="black",

#results %>% ggplot() + geom_hist(aes(x = simppval))
```

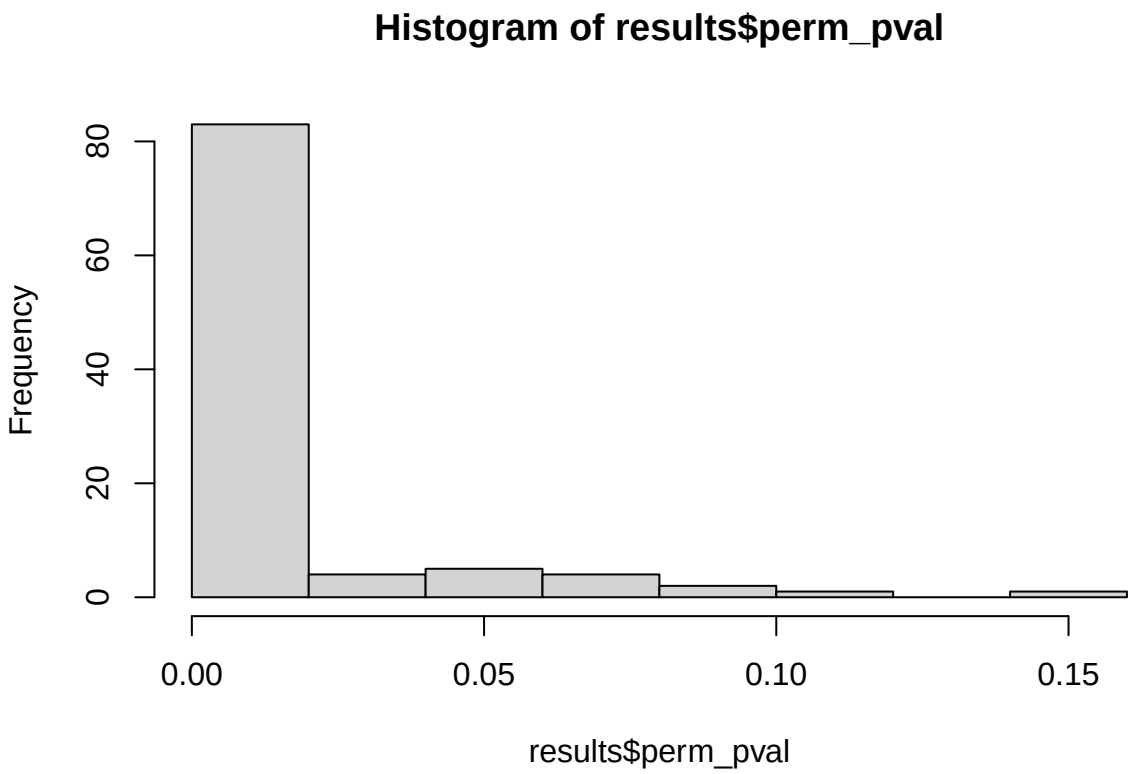
First, I'll correctly specify.

1.

```
#beta0,beta1,beta2,n_obs,n_data,n_shuffle
results<-LFunky(0,1,0,15,100,200)
hist(results$simpp_val)
```




```
hist(results$perm_pval)
```


Histogram of results$perm_pval
Frequency
0
20
40
60
80
0.00
0.05
0.10
0.15
results$perm_pval