Nonparametric Project of Agricultural Productivity in the U.S.

Permutational Tests & Spearman Correlation

Sofia Moroni*

2023-06-27

Contents


```
# National data
data_path = file.path('data')
output_path = file.path('results')
data =
    read.table(
        file.path(data_path, 'agricultural_indices.csv'),
        header = T,
        sep = ';'
    )

# Sostituzione delle virgole con punti
data<- data.frame(lapply(data, function(x) gsub(",", ".", x)))
data <- as.data.frame(lapply(data, as.numeric))
```

1 Load libraries and data

```
library(pbapply)
```

```
## Warning: il pacchetto 'pbapply' è stato creato con R versione 4.1.3
```

*sofia.moroni@mail.polimi.it

```r
library(dplyr)
```

```
## Warning: il pacchetto 'dplyr' è stato creato con R versione 4.1.3
```

```r
library(conformalInference)
library(ggplot2)
```

```
## Warning: il pacchetto 'ggplot2' è stato creato con R versione 4.1.3
```

```r
library(progress)
library(parallel)
library(coin)
```

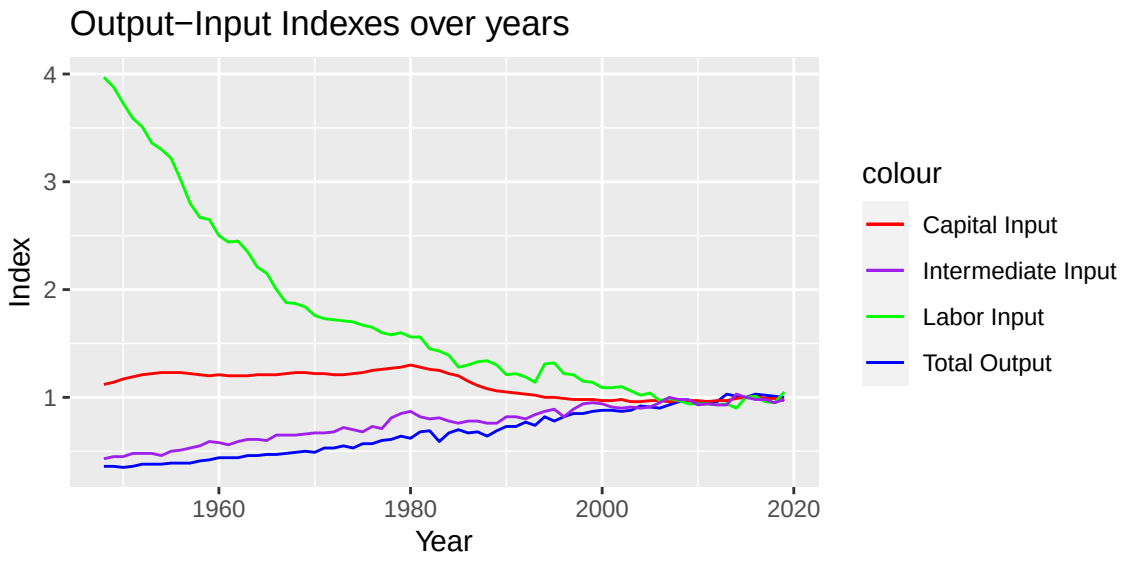
```
## Warning: il pacchetto 'coin' è stato creato con R versione 4.1.3
```

```
## Warning: il pacchetto 'survival' è stato creato con R versione 4.1.3
```

```r
# Set parameters
alpha <- 0.05
B <- 1000
seed <- 100
iter = 1000
```

2 Plot over years

```r
# Save only some variables
data_plots = data[,c(1,2,16,21,24)]

ggplot(data_plots, aes(x = Year)) +
  geom_line(aes(y = Total.agricultural.output, color = "Total Output")) +
  geom_line(aes(y = Capital.Total.Input, color = "Capital Input")) +
  geom_line(aes(y = Labor.Total.Input, color = "Labor Input")) +
  geom_line(aes(y = Intermediate.Total.Input, color = "Intermediate Input")) +
  labs(title = "Output-Input Indexes over years",
       x = "Year", y = "Index") +
  scale_color_manual(values = c("Total Output" = "blue", "Capital Input" = "red",
                                "Labor Input" = "green", "Intermediate Input" = "purple"))
```



3 Prepare data fo Permutational Tests

Capital Input

```
capital_input =
    read.table(
        file.path(data_path, 'capital_input.csv'),
        header = T,
        sep = ';'
    )

# Sostituzione delle virgole con punti
capital_input<- data.frame(lapply(capital_input, function(x) gsub(",", ".", x)))
capital_input <- as.data.frame(lapply(capital_input, as.numeric))
capital_input =t(capital_input)
colnames(capital_input) <- capital_input[1, ]
capital_input <- capital_input[-1, ]
capital_input = as.data.frame(capital_input)

pre_med <- apply(capital_input[,1:31], MARGIN = 1, FUN = mean)
post_med <- apply(capital_input[,32:45], MARGIN = 1, FUN = mean)

pre_data = data.frame(capital =pre_med)
post_data = data.frame(capital =post_med)
```

Labor Input

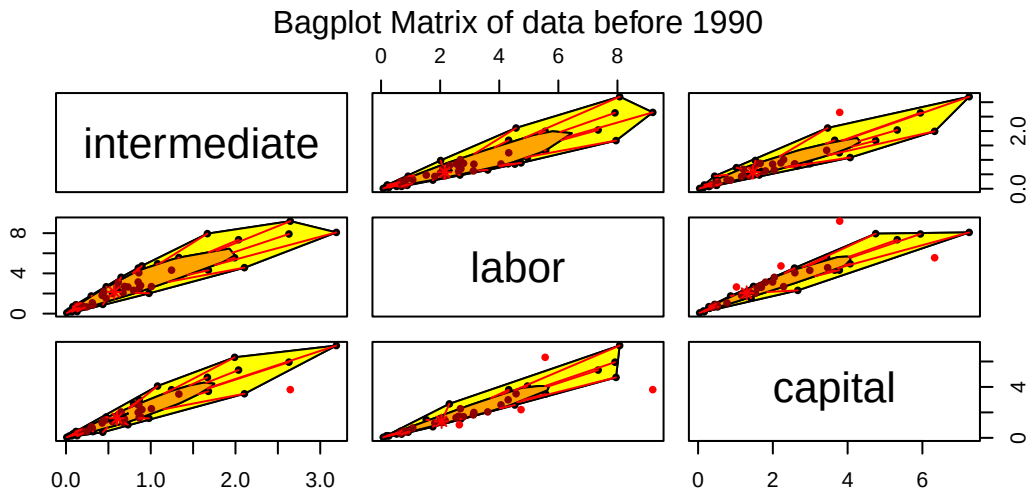
```
labor_input =
    read.table(
        file.path(data_path, 'labor_input_by_states.csv'),
        header = T,
        sep = ';'
    )
```

```r
# Sostituzione delle virgole con punti
labor_input<- data.frame(lapply(labor_input, function(x) gsub(",", ".", x)))
labor_input <- as.data.frame(lapply(labor_input, as.numeric))
labor_input =t(labor_input)
colnames(labor_input) <- labor_input[1, ]
labor_input <- labor_input[-1, ]
labor_input = as.data.frame(labor_input)

pre_med <- apply(labor_input[,1:31], MARGIN = 1, FUN = mean)
post_med <- apply(labor_input[,32:45], MARGIN = 1, FUN = mean)

pre_data = data.frame(labor =pre_med,pre_data)
post_data = data.frame(labor =post_med,post_data)
```

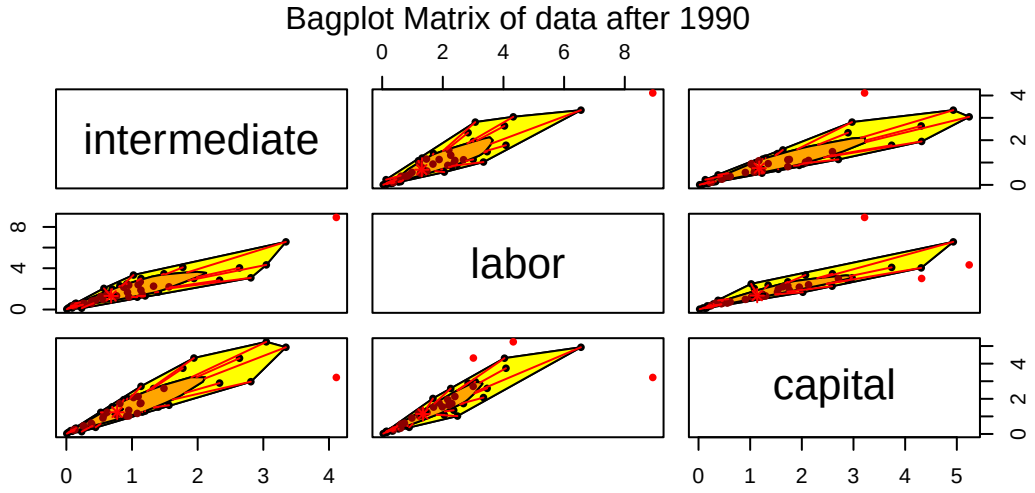
Intermediate Input

```r
intermediate_input =
    read.table(
        file.path(data_path, 'total_intermediate_input_by_states.csv'),
        header = T,
        sep = ';'
    )

# Sostituzione delle virgole con punti
intermediate_input<- data.frame(lapply(intermediate_input, function(x) gsub(",", ".", x)))
intermediate_input <- as.data.frame(lapply(intermediate_input, as.numeric))
intermediate_input =t(intermediate_input)
colnames(intermediate_input) <- intermediate_input[1, ]
intermediate_input <- intermediate_input[-1, ]
intermediate_input = as.data.frame(intermediate_input)
intermediate_input = intermediate_input[1:48,]

pre_med <- apply(intermediate_input[,1:31], MARGIN = 1, FUN = mean)
post_med <- apply(intermediate_input[,32:45], MARGIN = 1, FUN = mean)

pre_data = data.frame(intermediate =pre_med,pre_data)
post_data = data.frame(intermediate =post_med,post_data)
```

```r
library(ggplot2)
data_box = rbind(pre_data,post_data)

data_box$period[1:48] = 'first'
data_box$period[49:96] = 'second'



bagplot_matrix <- aplpack::bagplot.pairs(pre_data,main = "Bagplot Matrix of data before 1990", factor =
                col.looppoints = "black", col.baghull = "orange",
                col.bagpoints = "darkred")
```




```
bagplot_matrix <- aplpack::bagplot.pairs(post_data, main = "Bagplot Matrix of data after 1990",factor =
                  col.looppoints = "black", col.baghull = "orange",
                  col.bagpoints = "darkred")
```



Total Output

```
total_output =
    read.table(
        file.path(data_path, 'total_output_by_states.csv'),
        header = T,
        sep = ';'
    )

total_output = total_output[1:45,]
```

```r
# Sostituzione delle virgole con punti
total_output<- data.frame(lapply(total_output, function(x) gsub(",", ".", x)))
total_output <- as.data.frame(lapply(total_output, as.numeric))
total_output =t(total_output)
colnames(total_output) <- total_output[1, ]
total_output <- total_output[-1, ]
total_output = as.data.frame(total_output)
total_output = total_output[1:48,]

pre_med <- apply(total_output[,1:31], MARGIN = 1, FUN = mean)
post_med <- apply(total_output[,32:45], MARGIN = 1, FUN = mean)

total_output = data.frame(pre_output=pre_med, post_output=post_med)
```

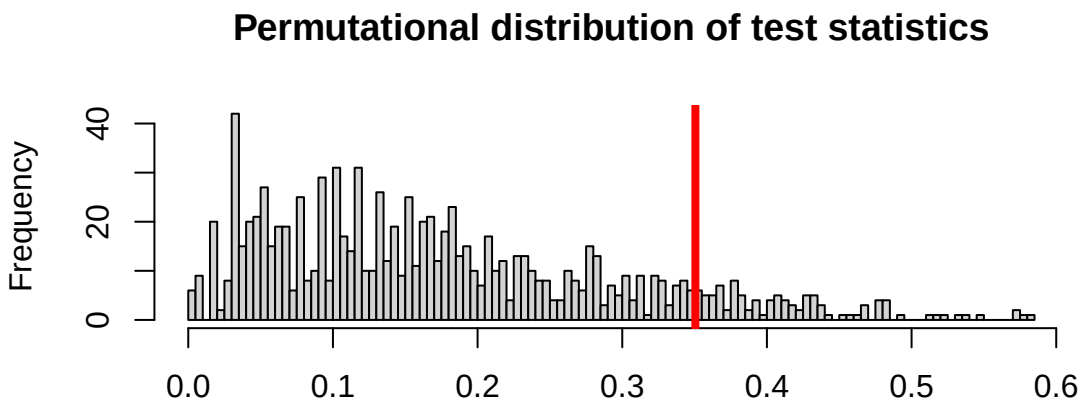
4 Permutational Tests

The following permutation tests were conducted using tables with state-level data. Specifically, the samples used consist of average values for the years 1960-1990 and 1990-2004 for each state.

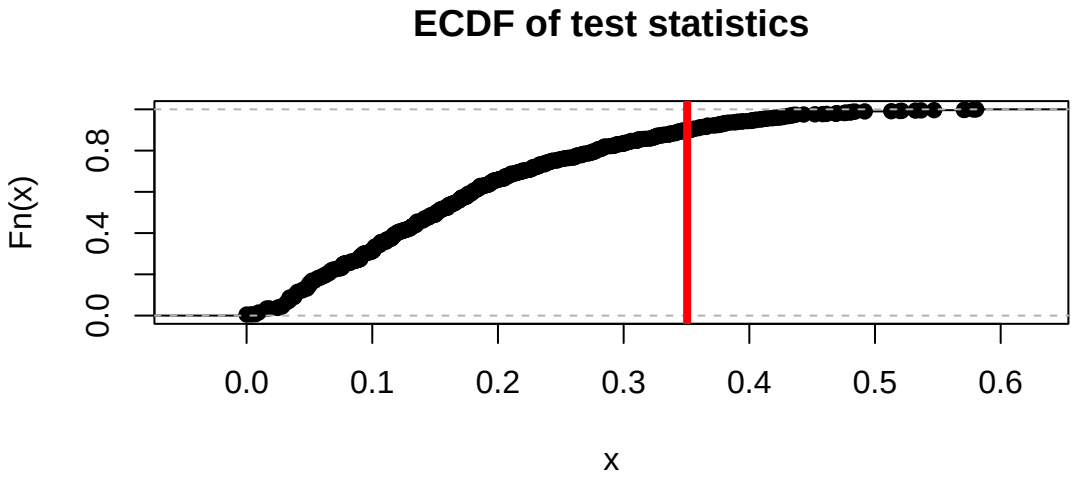
H_0: `Total Output_60_90` = `Total_Output_90_04` vs H_1: `Total_Output_60_90` $\neq$ `Total_Output_90_04`

```r
## TEST: H0:Y1=Y2 vs H1:Y1!=Y2 (in distribution) ##

x = total_output$pre_output
y = total_output$post_output

T0 <- abs(median(x) - median(y))
T_stat <- numeric(iter)
x_pooled <- c(x, y)
n <- length(x_pooled)
n1 <- length(x)
for (perm in 1:iter) {
        # permutation:

        permutation <- sample(1:n)
        x_perm <- x_pooled[permutation]
        x1_perm <- x_perm[1:n1]
        x2_perm <- x_perm[(n1 + 1):n]
        # test statistic:
        T_stat[perm] <- abs(median(x1_perm) - median(x2_perm))
    }
    # p-value
p_val <- sum(T_stat >= T0) / iter
```

```r
hist(sort(T_stat)[-1000],
     breaks = 100,
     main = 'Permutational distribution of test statistics',
     xlab = '')
abline(v = T0, col = 'red', lwd = 4)
```




```
plot(ecdf(sort(T_stat)[-1000]), main = 'ECDF of test statistics')
abline(v = T0, col = 'red', lwd = 4)
```




```
P = sum(T_stat >= T0) / B
P
```

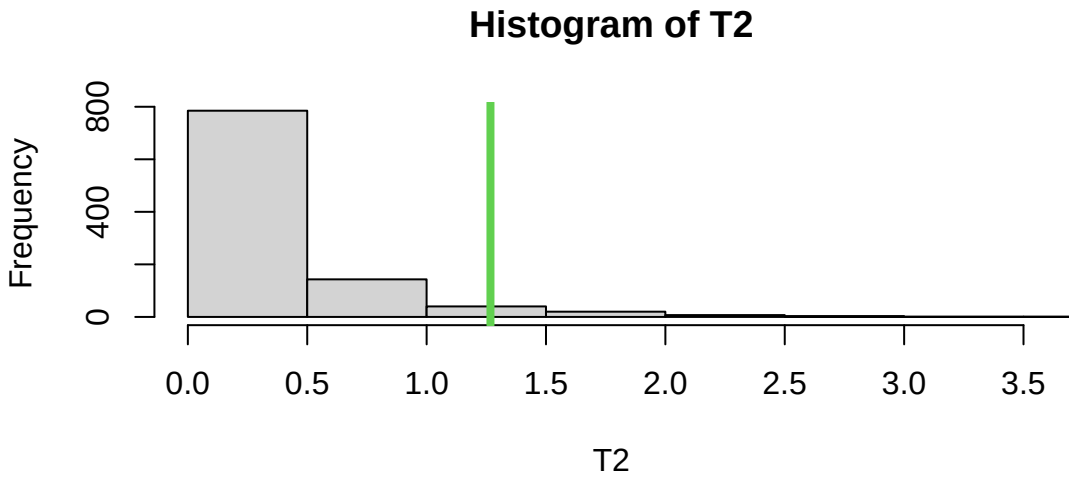
```
## [1] 0.102
```

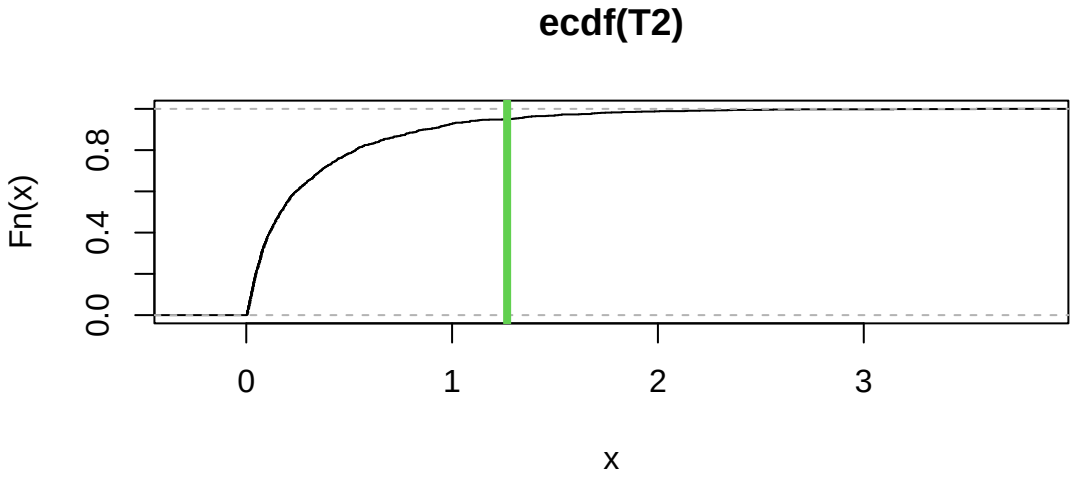
H_0: Inputs_60_90 $=$ Inputs_90_04 vs H_1: Inputs_60_90 $\neq$ Inputs_90_04

```
library(parallel)
library(pbapply)
```

```r
diagnostic_permutation <- function(T20, T2) {
    B <- length(T2)

    # Compare real test statistic with the ones given by the permuted data
    hist(T2, xlim = range(c(T2, T20)))
    abline(v = T20, col = 3, lwd = 4)

    # Empirical cumulative distribution function
    plot(ecdf(T2))
    abline(v = T20, col = 3, lwd = 4)

    # P-value
    p_val <- sum(T2 >= T20) / B
    cat("p-value: ", p_val)
}


compute_t_stat <- function(df1, df2) {
    df1.mean <- colMeans(df1)
    df2.mean <- colMeans(df2)
    return(norm(df1.mean - df2.mean, type='2')^2)
}

perm_wrapper = function(df1,df2) {
    df_pooled = rbind(df1, df2)
    n = nrow(df_pooled)
    n1 = nrow(df1)
    permutation = sample(n)
    df_perm = df_pooled[permutation, ]
    df1_perm = df_perm[1:n1, ]
    df2_perm = df_perm[(n1 + 1):n, ]
    compute_t_stat(df1_perm, df2_perm)
}
```

```r
# parallel
n_cores <- detectCores()
cl = makeCluster(n_cores)
invisible(clusterEvalQ(cl, library(DepthProc)))
clusterExport(cl, varlist = list("perm_wrapper", "pre_data", "post_data", "compute_t_stat"))
set.seed(100)
T2 <- pbreplicate(B, perm_wrapper(pre_data, post_data), cl = cl)
stopCluster(cl)

T20 = compute_t_stat(pre_data, post_data)

# diagnostic
diagnostic_permutation(T20, T2)
```






```
## p-value:  0.051
```

5 Spearman Correlation

```
#Input vs Total Output

#First Period
cor1_1 <- cor(data$Capital.Total.Input[1:43],data$Total.agricultural.output[1:43], method = "spearman")
cor1_2 <- cor(data$Labor.Total.Input[1:43],data$Total.agricultural.output[1:43], method = "spearman")
cor1_3 <- cor(data$Intermediate.Total.Input[1:43],data$Total.agricultural.output[1:43], method = "spearm
```

```r
#Second Period
cor2_1 <-cor(data$Capital.Total.Input[44:72],data$Total.agricultural.output[44:72], method = "spearman")
cor2_2 <-  cor(data$Labor.Total.Input[44:72],data$Total.agricultural.output[44:72], method = "spearman")
cor2_3 <-  cor(data$Intermediate.Total.Input[44:72],data$Total.agricultural.output[44:72], method = "sp
```

```r
my_matrix <- matrix(data = c(cor1_1,cor1_2,cor1_3,cor2_1,cor2_2,cor2_3), nrow = 2, ncol = 3, byrow = TRU

colnames(my_matrix) = c('Capital Input','Labor Input','Intermediate Input')
rownames(my_matrix) = c('Total Output 48-90','Total Output 90-19')
my_dataframe <- as.data.frame(my_matrix)

print(my_dataframe)
```

```
##                    Capital Input Labor Input Intermediate Input
## Total Output 48-90     0.1656040  -0.9876408          0.9568786
## Total Output 90-19    -0.3183959  -0.8513482          0.7858585
```

```r
cor1_1 <- cor(data$Capital.Durable.equipment.Input[1:43],data$Total.agricultural.output[1:43], method =
cor1_2 <- cor(data$Capital.Service.buildings.Input[1:43],data$Total.agricultural.output[1:43], method =
cor1_3 <- cor(data$Capital.Land.Input[1:43],data$Total.agricultural.output[1:43], method = "spearman")

cor2_1 <-cor(data$Capital.Durable.equipment.Input[44:72],data$Total.agricultural.output[44:72], method =
cor2_2 <-  cor(data$Capital.Service.buildings.Input[44:72],data$Total.agricultural.output[44:72], metho
cor2_3 <-  cor(data$Capital.Land.Input[44:72],data$Total.agricultural.output[44:72], method = "spearman
```