

# 1 Introduction

## 1.1 Background

This entry examines the variance, covariance, correlations of time to the initial response in milliseconds across a series of choice tasks. The R code shows how to compute Pearson and Spearman's correlations, variance, and covariance. Besides, this entry aims to explain these statistics' mathematical backgrounds at the same time. The data on initial response times were extracted from the 2016 predictive modeling competition in health preference research (Jakubczyk, Craig, et al. 2017).

In general, the total set of individuals that a researcher is interested in is called population. Ideally, we are able to measure all individuals, that is, the total population (measurements about a population are called **parameter**). However, in practice, it is often not possible to directly calculate parameter due to the costs of a research or methodological restrictions. Instead of parameter, on the other hand, **statistics** are measurements that are calculated based on sample data and aimed at predicting population's parameters. The following table is showing the correspondence between parameter (for a population) and statistics (for a sample).

We also examined this topic in the entry, *Illustrating time to initial response in choice tasks*, which explains the way of making histograms and tables and how to calculate the median and IQR of time to the initial response.

**Table 1.1.1: Parameters and Statistics**

	Parameters	Statistics
Mean ( $x_i$ )	$\mu$	$\bar{x}$
Variance	$\sigma_x^2, \sigma^2$	$s_x^2, s^2$
Standard Deviation	$\sigma_x, \sigma$	$s_x, s$
Covariance ( $x_i$ and $y_i$ )	$\sigma_{xy}, \sigma$	$s_{xy}, s$

Besides, in this entry, we also use the following notations:

$N$  = the total number of respondents (= the sample size )  
 $Z$  = the population size

## 1.2 Load libraries and source files

Notes: Change the **working directory** to the location of the source files on your computer. To do so, you need to replace the inside of `setwd()` with the location of data file on your computer.

```
setwd("C:\\Users\\aaa\\OneDrive\\USF\\Dr. Craig\\second entry")
# Please change the inside of quotes of 'setwd("")' to set your working directory.
library(knitr) # This is for 'kable,' which makes well-organized tables.
library(tidyverse) # This is for 'read_csv.'
library(magrittr) # This is for '%>%.'
library(ggplot2) # This is for 'ggplot.'
library(reshape2) # This is for 'dcast.'
library(corrplot) # This is for 'corrplot.'
library(dplyr) # This is for 'select.'
library(ggcorrplot) # This is for 'ggcorrplot.'
data1 <- read_csv("resp1wave1_220723.csv")
# By 'read_csv("")' you can load the data on initial response times.
```

We reorganize the loaded data, data1, which is a long format, into a wide format data, data2, by using the command, “dcast.”

```
data2 <- dcast(data1, survey_id ~ task)
# 'dcast' can reorganize data1 into a wide format, data2.
data2 <- data2[,-1]
# We remove the first column, which is for the IDs of each individual.
# This is because we have column names which are indicating individual IDs.
colnames(data2) <- paste0("Task", 1:max(data1$task)) # We change column names.
rownames(data2) <- paste0("ID ", 1:max(data1$survey_id)) # We change row names.
```

## 2 variance

### 2.1 Background

- *Definition:* The **probability mass function** for a discrete random variable,  $X$ , is given by  $f_X(x) = P(X = x)$ .
- *Definition:* The **Expectation** for  $g(X)$ , where  $X$  is a discrete random variable with the domain,  $\chi$ , is given by  $E[g(X)] = \sum_{x_i \in \chi} g(x_i)f_X(x_i)$ .
- *Definition:* The **(unbiased) sample variance**,  $Var(X)$  or  $s_x^2$ , shows the dispersion of the random variable and is given by  $Var(X) = E[(X - \mu)^2]$  or  $s_x^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$

When it comes to a sample variance, there are two types; **the unbiased estimator for population variance (the unbiased sample variance)** and **the biased estimator for population variance (the biased sample variance)**. In general, when people refer to the sample variance, they are most likely referring to the unbiased sample variance, which is often denoted by  $s_x^2$ . To emphasize the difference, let  $s_x^2$  denote the unbiased sample variance and let  $b_x^2$  denote the biased sample variance.

$$s_x^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (2.1.1)$$

$$b_x^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (2.1.2)$$

Since the sample size is  $N$ , Equation (2.1.2) may look more natural as an estimator for the population variance,  $\sigma_x^2$ .

$$\sigma_x^2 = \frac{1}{Z} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (2.1.3)$$

However, we have the following fact shown in Equation (2.1.4).

$$E[b_x^2] = \frac{N-1}{N} \sigma_x^2 \neq \sigma_x^2 \quad (2.1.4)$$

As a property of the unbiasedness, the expectation of the sample variance is required to correspond to the population variance. The adjustment is shown in Equation (2.1.5) and thus, we get the unbiased sample variable,  $s_x^2 = \frac{N}{N-1} \cdot b_x^2$ . That is, the expectation of the sample variance,  $s_x^2$ , is equal to the population variance,  $\sigma_x^2$ .

$$E\left[\frac{N}{N-1} \cdot b_x^2\right] = \frac{N}{N-1} \cdot \left\{ \frac{N-1}{N} \sigma_x^2 \right\} = \sigma_x^2 \quad (2.1.5)$$

$$E[s_x^2] = \sigma_x^2 \quad (2.1.6)$$

## 2.2 Example I

The following calculation is made based on Equation (2.1.1).

```
frac <- function (x) {return(x/(nrow(data2)-1))}
# Here we define a new function, frac, which divides input, x, by "N-1"
v2 = NULL
for (i in 1:20) {
  v1 <- {data2[,i]-mean(data2[,i])}^2 %>% sum() %>% frac()
  v2 <- cbind(v2, v1)
}# For each column, that is, for each task, we calculate variance here.
# This process is done by repeating the same calculation to each task by using 'for.'
v3 <- as.data.frame(v2) # To make v2 as a data-frame.
colnames(v3) = paste0("Task", 1:20) # This changes the name of columns.
rownames(v3) = c("Variance") # This changes the name of rows.
v3 <- as.matrix(v3)
v3 <- formatC(v3, digits = 2, format = "e", ) %>% as.data.frame()
# By using 'formatC,' we can reduce significant digits.
# To do so, we cannot input lists into 'formatC.'
# This is why we convert the data.frame, 'cov3,' to the matrix, 'cov4'
v4 <- select(.data = v3, -c(11:20)) # By using 'select,' we separate the tables.
v5 <- select(.data = v3, -c(1:10)) # By using 'select,' we separate the tables.
kable(v4, caption = "**Table 2.2: Variance of the initial response time for Task 1 to 20**")
```

Table 2.2: Variance of the initial response time for Task 1 to 20

	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10
Variance	1.14e+09	1.05e+09	1.06e+09	9.96e+08	9.68e+08	9.92e+08	1.13e+09	1.06e+09	9.44e+08	9.70e+08

```
# By using 'kable,' we can make a table that could be exported to PDF or HTML files.
kable(v5) # The second part of the same table since we separated it above.
```

	Task11	Task12	Task13	Task14	Task15	Task16	Task17	Task18	Task19	Task20
Variance	1.31e+09	9.24e+08	5.91e+08	8.69e+08	7.45e+08	6.86e+08	5.95e+08	6.53e+08	6.60e+08	5.51e+08

## 3 covariance

### 3.1 Background

- *Definition:* The **covariance** is given by  $Cov(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]$  or  $s_{xy} = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})$ .

$$s_{xy} = \underbrace{\frac{1}{N-1}}_{(*3)} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y}) \quad (3.1.1)$$

(\*3) in Equation (3.1.1) is given by the same logic in the argument of the sample variance.

## 3.2 Example I

The following calculation is made based on Equation (3.1.1).

```

cov1 <- function (i,j) {
  cov0 <- {(data2[,i]-mean(data2[,i]))*(data2[,j]-mean(data2[,j]))} %>% sum() %>% frac()
  return(cov0)
} # We defined a new function, cov1.
# 'cov1' returns covariance of two columns by inputting two number of columns.
lis1 <- rep(c(1:20),each=20)
# This is a list that repeats each of 20 numbers 20 times such that,
# "1"*20, "2"*20, ..., "20"*20.
lis2 <- rep(c(1:20),20)
# This is a list that repeats the sequence 1-20 twenty times such that,
# "1, 2, ..., 20"*20.
# The lists, lis1 and lis2, are used for appointing two columns
# that would be input into functions.
cov2 <- mapply(FUN=cov1, i=lis1, j=lis2) %>% as.data.frame()
# By using 'mapply,' we can input the two lists, lis1 and lis2, into the function, 'cov1.'
# As a result, we get the list of covariances for all combinations of the 20 tasks.
cov2 <- cbind(lis1, lis2, cov2)
# This combines the two lists and the result, 'cov2,' into one matrix.
colnames(cov2) <- c("lis1", "lis2", "covariance")
# We change the name of column names for, 'cov2.'
cov3 <- dcast(cov2, lis1 ~ lis2, value.var = "covariance" )
# By using 'dcast,' We can reorganize a long format data, 'cov2,'
# and get a wide format data, 'cov3.'
cov4 <- as.matrix(cov3)
cov5 <- formatC(cov4, digits = 2, format = "e", )
cov5 <- cov5[,-1] %>% as.data.frame()
# By using 'formatC,' we can reduce significant digits.
# To do so, we cannot input list into 'formatC.'
# Thus, we convert 'cov3' to a matrix, 'cov4'
colnames(cov5) <- paste0("Task", 1:20)
rownames(cov5) <- paste0("Task", 1:20)
# In the two lines above, we change the column and row names of 'cov5'.
cov6 <- select(.data = cov5, -c(11:20))
cov7 <- select(.data = cov5, -c(1:10))
# As we did in the section of variance, we separate the data, 'cov5,' and
# obtain two data, 'cov6' and 'cov7.'
kable(cov6, caption = "**Table 3.2: Covariance of the initial response time for Task 1 to 20**")

```

**Table 3.2: Covariance of the initial response time for Task 1 to 20**

	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10
Task1	1.14e+09	3.18e+08	2.87e+08	2.21e+08	2.56e+08	1.88e+08	2.03e+08	2.47e+08	1.87e+08	2.10e+08
Task2	3.18e+08	1.05e+09	2.58e+08	2.89e+08	2.73e+08	2.34e+08	2.44e+08	2.63e+08	2.56e+08	2.32e+08
Task3	2.87e+08	2.58e+08	1.06e+09	2.62e+08	2.90e+08	2.04e+08	2.79e+08	1.62e+08	1.95e+08	2.42e+08
Task4	2.21e+08	2.89e+08	2.62e+08	9.96e+08	2.47e+08	2.20e+08	2.40e+08	2.47e+08	2.10e+08	2.06e+08
Task5	2.56e+08	2.73e+08	2.90e+08	2.47e+08	9.68e+08	2.46e+08	2.65e+08	2.97e+08	2.41e+08	2.16e+08
Task6	1.88e+08	2.34e+08	2.04e+08	2.20e+08	2.46e+08	9.92e+08	2.57e+08	2.31e+08	2.53e+08	2.37e+08
Task7	2.03e+08	2.44e+08	2.79e+08	2.40e+08	2.65e+08	2.57e+08	1.13e+09	3.26e+08	2.86e+08	2.44e+08
Task8	2.47e+08	2.63e+08	1.62e+08	2.47e+08	2.97e+08	2.31e+08	3.26e+08	1.06e+09	2.57e+08	2.48e+08
Task9	1.87e+08	2.56e+08	1.95e+08	2.10e+08	2.41e+08	2.53e+08	2.86e+08	2.57e+08	9.44e+08	2.26e+08
Task10	2.10e+08	2.32e+08	2.42e+08	2.06e+08	2.16e+08	2.37e+08	2.44e+08	2.48e+08	2.26e+08	9.70e+08
Task11	2.35e+08	2.83e+08	2.68e+08	2.36e+08	2.66e+08	1.98e+08	2.33e+08	2.66e+08	2.41e+08	2.56e+08
Task12	1.66e+08	1.68e+08	1.63e+08	1.65e+08	1.63e+08	1.82e+08	1.95e+08	1.93e+08	2.03e+08	1.86e+08
Task13	1.31e+08	1.38e+08	1.44e+08	1.06e+08	1.25e+08	1.24e+08	1.67e+08	1.73e+08	1.64e+08	1.71e+08
Task14	1.33e+08	1.45e+08	1.11e+08	1.42e+08	1.35e+08	1.33e+08	1.67e+08	1.41e+08	1.91e+08	1.54e+08
Task15	8.98e+07	1.26e+08	1.51e+08	1.30e+08	1.53e+08	1.37e+08	1.59e+08	1.13e+08	1.20e+08	1.31e+08
Task16	1.54e+08	1.13e+08	1.18e+08	1.34e+08	1.29e+08	1.42e+08	1.59e+08	1.23e+08	1.50e+08	1.25e+08
Task17	1.30e+08	1.02e+08	1.11e+08	9.48e+07	9.01e+07	1.10e+08	8.08e+07	1.17e+08	1.31e+08	9.14e+07
Task18	1.15e+08	1.10e+08	1.00e+08	1.10e+08	1.05e+08	1.03e+08	1.15e+08	1.65e+08	1.21e+08	7.71e+07
Task19	8.84e+07	1.01e+08	1.14e+08	9.00e+07	1.64e+08	1.02e+08	1.09e+08	1.87e+08	1.35e+08	9.91e+07
Task20	8.01e+07	1.19e+08	1.33e+08	1.03e+08	9.73e+07	1.02e+08	1.16e+08	1.22e+08	1.17e+08	7.49e+07

*# By 'kable,' we can make a good-looking table.*  
kable(cov7)

	Task11	Task12	Task13	Task14	Task15	Task16	Task17	Task18	Task19	Task20
Task1	2.35e+08	1.66e+08	1.31e+08	1.33e+08	8.98e+07	1.54e+08	1.30e+08	1.15e+08	8.84e+07	8.01e+07
Task2	2.83e+08	1.68e+08	1.38e+08	1.45e+08	1.26e+08	1.13e+08	1.02e+08	1.10e+08	1.01e+08	1.19e+08
Task3	2.68e+08	1.63e+08	1.44e+08	1.11e+08	1.51e+08	1.18e+08	1.11e+08	1.00e+08	1.14e+08	1.33e+08
Task4	2.36e+08	1.65e+08	1.06e+08	1.42e+08	1.30e+08	1.34e+08	9.48e+07	1.10e+08	9.00e+07	1.03e+08
Task5	2.66e+08	1.63e+08	1.25e+08	1.35e+08	1.53e+08	1.29e+08	9.01e+07	1.05e+08	1.64e+08	9.73e+07
Task6	1.98e+08	1.82e+08	1.24e+08	1.33e+08	1.37e+08	1.42e+08	1.10e+08	1.03e+08	1.02e+08	1.02e+08
Task7	2.33e+08	1.95e+08	1.67e+08	1.67e+08	1.59e+08	1.59e+08	8.08e+07	1.15e+08	1.09e+08	1.16e+08
Task8	2.66e+08	1.93e+08	1.73e+08	1.41e+08	1.13e+08	1.23e+08	1.17e+08	1.65e+08	1.87e+08	1.22e+08
Task9	2.41e+08	2.03e+08	1.64e+08	1.91e+08	1.20e+08	1.50e+08	1.31e+08	1.21e+08	1.35e+08	1.17e+08
Task10	2.56e+08	1.86e+08	1.71e+08	1.54e+08	1.31e+08	1.25e+08	9.14e+07	7.71e+07	9.91e+07	7.49e+07
Task11	1.31e+09	2.58e+08	2.10e+08	1.83e+08	1.43e+08	1.83e+08	1.95e+08	1.30e+08	1.58e+08	1.09e+08
Task12	2.58e+08	9.24e+08	2.51e+08	2.01e+08	1.95e+08	1.47e+08	1.06e+08	9.63e+07	1.32e+08	1.02e+08
Task13	2.10e+08	2.51e+08	5.91e+08	1.67e+08	1.52e+08	1.36e+08	1.24e+08	7.05e+07	8.70e+07	7.78e+07
Task14	1.83e+08	2.01e+08	1.67e+08	8.69e+08	2.42e+08	1.83e+08	1.14e+08	1.12e+08	1.10e+08	1.39e+08
Task15	1.43e+08	1.95e+08	1.52e+08	2.42e+08	7.45e+08	1.57e+08	1.15e+08	1.52e+08	1.02e+08	1.07e+08
Task16	1.83e+08	1.47e+08	1.36e+08	1.83e+08	1.57e+08	6.86e+08	1.36e+08	1.05e+08	1.24e+08	9.81e+07
Task17	1.95e+08	1.06e+08	1.24e+08	1.14e+08	1.15e+08	1.36e+08	5.95e+08	1.20e+08	1.05e+08	1.10e+08
Task18	1.30e+08	9.63e+07	7.05e+07	1.12e+08	1.52e+08	1.05e+08	1.20e+08	6.53e+08	1.48e+08	1.11e+08
Task19	1.58e+08	1.32e+08	8.70e+07	1.10e+08	1.02e+08	1.24e+08	1.05e+08	1.48e+08	6.60e+08	1.50e+08
Task20	1.09e+08	1.02e+08	7.78e+07	1.39e+08	1.07e+08	9.81e+07	1.10e+08	1.11e+08	1.50e+08	5.51e+08

## 4 Pearson correlation

### 4.1 Background

Let  $x_i$  and  $y_i$  be the initial response time for one of the twenty tasks ( $1 \leq i \leq N$ ). Pearson correlation,  $r_p$ , is given by Equation (4.1.1).

$$r_p = \frac{s_{xy}}{s_x s_y} = \frac{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2} \sqrt{\frac{1}{N-1} \sum_{i=1}^N (y_i - \bar{y})^2}} = \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^N (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^N (y_i - \bar{y})^2}} \quad (4.1.1)$$

Pearson correlation is the most common correlation coefficient. When people simply mention the correlation coefficient between random variables X and Y,  $Cor(X, Y)$ , they are likely referring to pearson correlation. Pearson correlation could capture the linear relation between two random variables. As its property,  $r_p$  satisfies  $-1 \leq r_p \leq 1$ .

When  $r_p = 1$ , its means there exists a perfect positive linear relation, while  $r_p = -1$ , its means there exists a perfect negative linear relation. When  $r_p = 0$ , there is no association between two variables.

The closer  $|r_p|$  reaches 1, the stronger the correlation is. There is no absolutely right criteria, but we can roughly say the following;

**Table 4.1.1: The value of Correlation Coefficient and its indication**

The value of Correlation Coefficient	Description	The value of Correlation Coefficient	Description
1.0	Perfect positive linear relation	-1.0	Perfect linear negative relation
[0.8, 1.0)	Very strong positive relation	[-0.8, -1.0)	Very strong negative relation
[0.6, 0.8)	Strong positive relation	[-0.6, -0.8)	Strong negative relation
[0.4, 0.6)	Moderate positive relation	[-0.4, -0.6)	Moderate negative relation
[0.2, 0.4)	Weak positive relation	[-0.2, -0.4)	Weak negative relation
[0.0, 0.2)	Very weak positive relation / no relation	[0.0, -0.2)	Very weak negative relation / no relation

### 4.2 Example I

The following calculation is made based on Equation (4.1.1).

```
meanall <- data.frame(apply(data2, 2, mean))
# By using 'apply,' we can apply a function to each column of a matrix.
colnames(meanall) <- c("mean") # We change the name of column names.
meanall <- as.data.frame(t(meanall)) # We exchange the row and column of 'meanall.'
fun1 <- function(i,j){
  dif1 <- data2[,i]-rep(meanall[,i], max(data1$survey_id))
  dif2 <- data2[,j]-rep(meanall[,j], max(data1$survey_id))
  a <- sum(dif1 * dif2)/((sum((dif1)^2)*sum((dif2)^2))^0.5)
  return(a)
}
# We defined a new function, fun1, which returns a pearson correlation by inputting
```

```

# two column numbers.
result <- mapply(FUN=fun1, i=lis1, j=lis2, MoreArgs = NULL,
                SIMPLIFY = TRUE, USE.NAMES = TRUE) %>%
  as.data.frame() %>% cbind(lis1, lis2)
# By using 'mapply,' we can input the two lists, lis1 and lis2, into the function, 'fun1.'
# As a result, we get the list of pearson correlations for all combinations of the 20 tasks.
colnames(result) <- c("cor", "lis1", "lis2") # We change column names.
r3 <- dcast(result, lis2 ~ lis1, value.var = "cor")
# By using 'dcast,' We can reorganize a long format data, 'result,'
# and get a wide format data, 'r3.'
r3 <- r3[,-1] # We remove the first column that is showing IDs.
colnames(r3) <- paste0("Task", 1:20)
rownames(r3) <- paste0("Task", 1:20)
# In the two lines above, we change the column and row names of 'r3.'
r3<- round(r3, 3)# By using 'round,' we can change significant digits.
# We reduce the significant digits of each entry of 'r3' to three decimals.
r4 <- select(.data = r3, -c(11:20))
r5 <- select(.data = r3, -c(1:10))
# As we did before, by using 'select,' we separate the data, 'r3,'
# and obtain into two data, 'r4' and 'r5'
kable(r4, align = "c",
      caption = "***Table 4.2.1: Pearson Correlations of the initial response time for Task 1 to 20**",
      row.names = T, escape = FALSE, centering = T)

```

**Table 4.2.1: Pearson Correlations of the initial response time for Task 1 to 20**

	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10
Task1	1.000	0.290	0.260	0.207	0.244	0.177	0.178	0.225	0.180	0.199
Task2	0.290	1.000	0.244	0.282	0.271	0.229	0.224	0.250	0.256	0.230
Task3	0.260	0.244	1.000	0.255	0.286	0.199	0.255	0.154	0.195	0.239
Task4	0.207	0.282	0.255	1.000	0.251	0.221	0.226	0.241	0.217	0.209
Task5	0.244	0.271	0.286	0.251	1.000	0.251	0.253	0.293	0.253	0.223
Task6	0.177	0.229	0.199	0.221	0.251	1.000	0.242	0.225	0.262	0.242
Task7	0.178	0.224	0.255	0.226	0.253	0.242	1.000	0.298	0.276	0.233
Task8	0.225	0.250	0.154	0.241	0.293	0.225	0.298	1.000	0.257	0.245
Task9	0.180	0.256	0.195	0.217	0.253	0.262	0.276	0.257	1.000	0.236
Task10	0.199	0.230	0.239	0.209	0.223	0.242	0.233	0.245	0.236	1.000
Task11	0.192	0.241	0.227	0.206	0.236	0.174	0.191	0.226	0.216	0.227
Task12	0.161	0.170	0.164	0.172	0.172	0.190	0.190	0.196	0.218	0.196
Task13	0.159	0.175	0.181	0.138	0.166	0.162	0.204	0.220	0.219	0.226
Task14	0.134	0.152	0.116	0.153	0.147	0.143	0.169	0.147	0.211	0.167
Task15	0.097	0.142	0.170	0.151	0.180	0.160	0.174	0.127	0.143	0.154
Task16	0.174	0.133	0.138	0.162	0.159	0.172	0.181	0.144	0.186	0.154
Task17	0.158	0.130	0.140	0.123	0.119	0.143	0.098	0.147	0.175	0.120
Task18	0.133	0.133	0.120	0.136	0.132	0.129	0.134	0.198	0.155	0.097
Task19	0.102	0.122	0.136	0.111	0.205	0.126	0.126	0.224	0.171	0.124
Task20	0.101	0.157	0.174	0.139	0.133	0.138	0.147	0.160	0.162	0.102

```

# By using 'kable,' we make a table.
kable(r5)

```

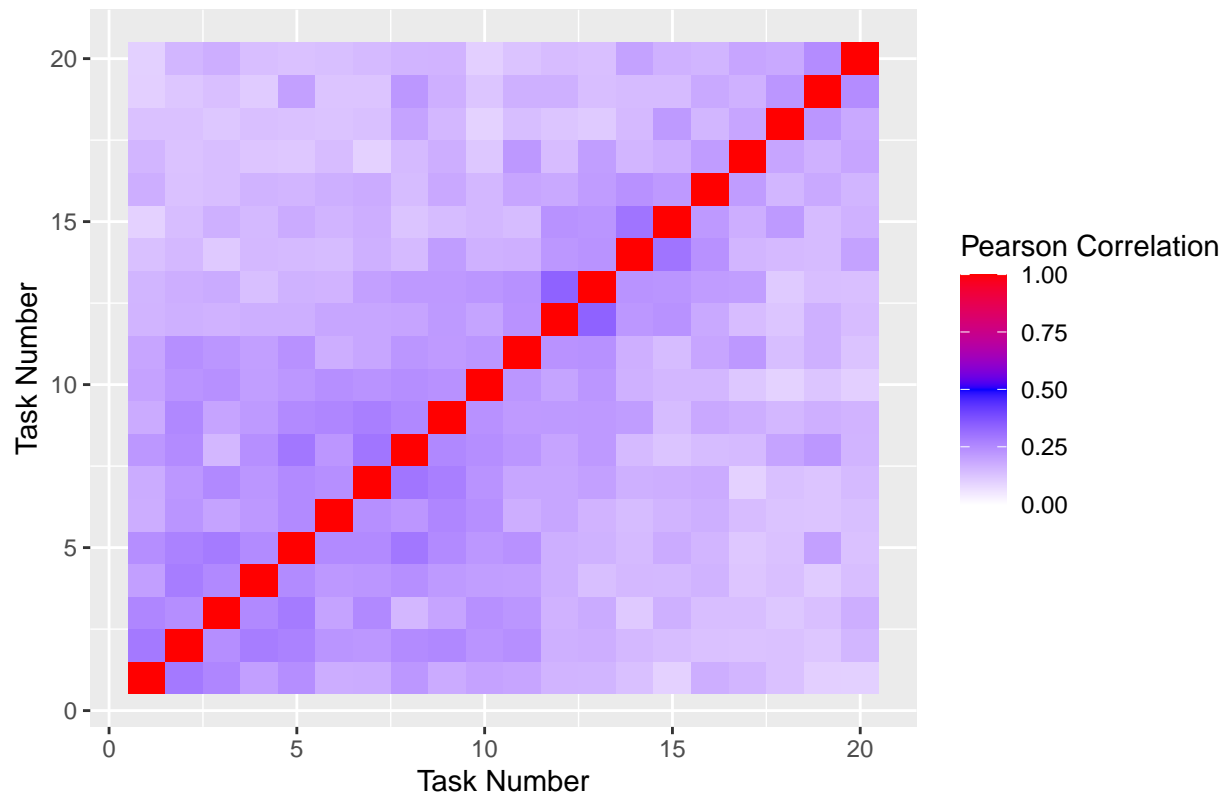
	Task11	Task12	Task13	Task14	Task15	Task16	Task17	Task18	Task19	Task20
Task1	0.192	0.161	0.159	0.134	0.097	0.174	0.158	0.133	0.102	0.101
Task2	0.241	0.170	0.175	0.152	0.142	0.133	0.130	0.133	0.122	0.157
Task3	0.227	0.164	0.181	0.116	0.170	0.138	0.140	0.120	0.136	0.174
Task4	0.206	0.172	0.138	0.153	0.151	0.162	0.123	0.136	0.111	0.139
Task5	0.236	0.172	0.166	0.147	0.180	0.159	0.119	0.132	0.205	0.133
Task6	0.174	0.190	0.162	0.143	0.160	0.172	0.143	0.129	0.126	0.138
Task7	0.191	0.190	0.204	0.169	0.174	0.181	0.098	0.134	0.126	0.147
Task8	0.226	0.196	0.220	0.147	0.127	0.144	0.147	0.198	0.224	0.160
Task9	0.216	0.218	0.219	0.211	0.143	0.186	0.175	0.155	0.171	0.162
Task10	0.227	0.196	0.226	0.167	0.154	0.154	0.120	0.097	0.124	0.102
Task11	1.000	0.234	0.238	0.171	0.144	0.192	0.221	0.140	0.169	0.128
Task12	0.234	1.000	0.340	0.224	0.236	0.184	0.143	0.124	0.169	0.142
Task13	0.238	0.340	1.000	0.233	0.229	0.214	0.209	0.113	0.139	0.136
Task14	0.171	0.224	0.233	1.000	0.300	0.237	0.159	0.149	0.145	0.201
Task15	0.144	0.236	0.229	0.300	1.000	0.219	0.173	0.218	0.145	0.167
Task16	0.192	0.184	0.214	0.237	0.219	1.000	0.213	0.157	0.185	0.160
Task17	0.221	0.143	0.209	0.159	0.173	0.213	1.000	0.193	0.167	0.193
Task18	0.140	0.124	0.113	0.149	0.218	0.157	0.193	1.000	0.226	0.185
Task19	0.169	0.169	0.139	0.145	0.145	0.185	0.167	0.226	1.000	0.248
Task20	0.128	0.142	0.136	0.201	0.167	0.160	0.193	0.185	0.248	1.000

To show the results of calculation above in a heatmap, we can use ‘ggplot.’ The output is Figure 4.2.2.

```
ggplot(result, aes(x= lis1, y = lis2, fill = cor)) +
  geom_tile() +
  scale_fill_gradient2(low = "white", high = "red", mid = "blue",
    midpoint = 0.5, limit = c(0,1),
    name = "Pearson Correlation", space = "Lab") +
  labs(x = "Task Number", y = "Task Number",
    title = "Figure 4.2.2: Pearson Correlation of the initial response time")
```



Figure 4.2.2: Pearson Correlation of the initial response time

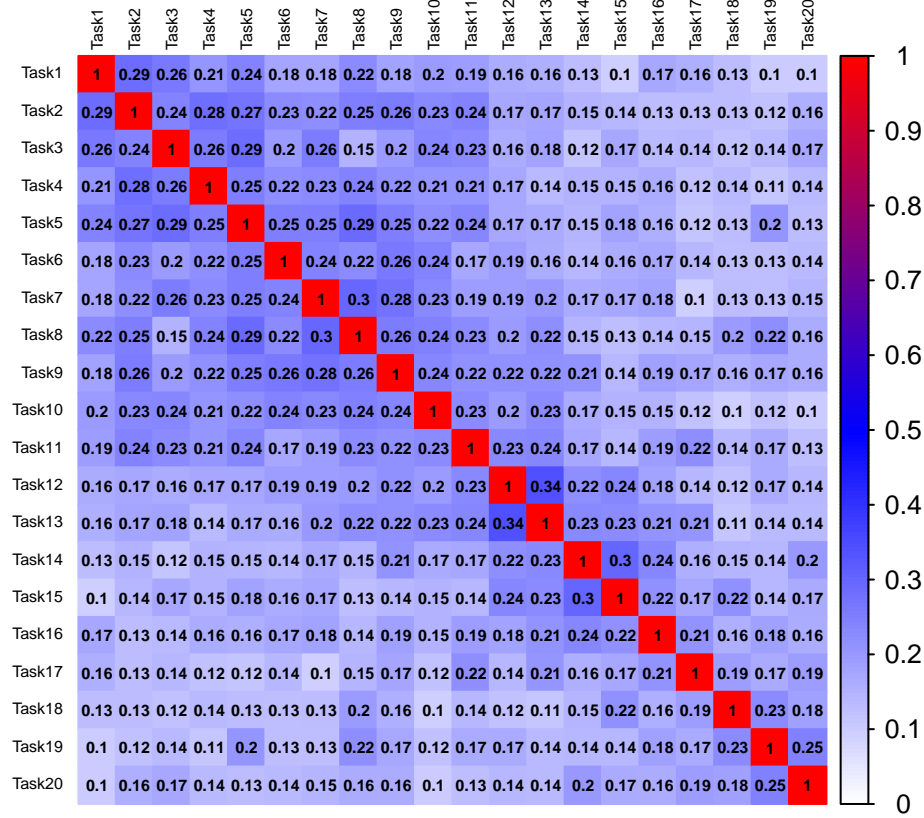


```
# By using 'ggplot,' we can make a heat-map from the long format data, 'result,'
# which stores pearson correlations for all combinations of the 20 tasks.
# As shown in 'aes(x= lis1, y = lis2, fill = cor),' it is necessary to indicate
# which columns of 'result' are x-axis, y-axis, and data contents ('fill') respectively.
# By 'scale_fill_gradient2,' we can change the format of the figure.
# By 'labs,' we can add labels to x-axis, y-axis, and the title of the figure.
```

To add coefficient on a figure, we can use 'corrplot' shown below.

```
r4 <- as.matrix(r3) # we convert the data.frame, 'r3,' to the matrix, 'r4.'
corrplot(r4, method = "color", is.corr = FALSE, addCoef.col="black",
         tl.cex = 0.5, number.cex = 0.5, tl.col = "black",
         col.lim = c(0, 1),
         col=colorRampPalette(c("white","blue","red"))(200),
         title = "Figure 4.2.3: Pearson Correlation of the initial response time",
         mar=c(0,0,1,0) )
```

**Figure 4.2.3: Pearson Correlation of the initial response time**



# By using 'corrplot,' we can make a heat-map with the indications of coefficients.  
# We only can input numeric but cannot input list into 'corrplot.'  
# This is why we convert the data.frame, 'r3,' to the matrix, 'r4.'

## 5 Spearman correlation

### 5.1 Background

Spearman's rank correlation coefficient is sharing the basic idea with Pearson correlation. The difference is that we use ranks of individual value instead of actual observed values. So, if the sample size is  $N$ , that is, there are  $N$  of individual values for one variable, then the individual values will be assigned from 1 to  $N$  ranks in terms of the magnitude of value for the variable.

As its feature, it is noted that Spearman's correlation can capture a nonlinear relation between two variables in contrast to Pearson correlation that captures a linear relation. More precisely, Spearman's correlation cannot implicitly capture complicated relations such that, relations shown in squares, cubes, exponent, and so on. Instead, it is able to tell us if there is a correlation of increases in both variables or decreases in both at any time, which is called a monotonic relation. In other words, given two variables,  $x_1$ ,  $x_2$ , and a function s.t.  $x_2 = f(x_1)$ , Spearman's correlation gives a guess as to if we have  $f'(x_1) \leq 0$  or  $f'(x_1) \geq 0$  for  $\forall x_1$ .

Let  $R_i$  be the rank of  $x_i$  and  $R'_i$  be the rank of  $y_i$ . Then Spearman's rank correlation,  $r_s$ , is given by Equation (5.1.1). As its property,  $r_s$  satisfies  $-1 \leq r_s \leq 1$ . Table 4.1.1, which tells us the indication of the value of correlation, is still valid for Spearman's rank correlation.

$$r_s = 1 - \frac{6 \sum_{i=1}^N d_i^2}{N(N^2 - 1)} \quad (5.1.1)$$

where

$r_s$  = Spearman's rank correlation coefficient

$$d_i^2 = (R_i - R'_i)^2$$

Equation (5.1.1) could be derived by Equation (4.1.1) by using the following four facts;

$$1. \sum_{i=1}^N R_i = \sum_{i=1}^N R'_i = 1 + 2 + \dots + N = \frac{N(N+1)}{2} \quad (5.1.2)$$

$$2. \sum_{i=1}^N R_i^2 = \sum_{i=1}^N R_i'^2 = 1^2 + 2^2 + \dots + N^2 = \frac{N(N+1)(2N+1)}{6} \quad (5.1.3)$$

$$3. \bar{R}_i = \bar{R}'_i = \frac{1}{N} \sum_{i=1}^N R_i = \frac{1}{N} \sum_{i=1}^N R'_i = \frac{1}{N} \frac{N(N+1)}{2} = \frac{N+1}{2} \quad (5.1.4)$$

$$4. \bar{R}_i^2 = \bar{R}'_i^2 = \left(\frac{N+1}{2}\right)^2 = \frac{(N+1)^2}{4} \quad (5.1.5)$$

(1)

By definition of Pearson correlation, we know the following;

$$r_p = r_{xy} = \frac{S_{xy}}{S_x S_y} = \frac{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2} \sqrt{\frac{1}{N-1} \sum_{i=1}^N (y_i - \bar{y})^2}} = \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^N (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^N (y_i - \bar{y})^2}}$$

We replace  $x_i$  with  $R_i$  and  $y_i$  with  $R'_i$  to convert Pearson correlation into Spearman correlation.

$$r_s = \frac{\frac{1}{N-1} \sum_{i=1}^N (R_i - \bar{R})(R'_i - \bar{R}')}{\sqrt{\left[\frac{1}{N-1} \sum_{i=1}^N (R_i - \bar{R})\right] \left\{\frac{1}{N-1} \sum_{i=1}^N (R'_i - \bar{R}')\right\}}} = \frac{\sum_{i=1}^N (R_i - \bar{R})(R'_i - \bar{R}')}{\sqrt{\left[\sum_{i=1}^N (R_i - \bar{R})\right] \left\{\sum_{i=1}^N (R'_i - \bar{R}')\right\}}} \quad (5.1.6)$$

Now we consider  $d_i^2$ .

$$\begin{aligned} d_i^2 &= (R_i - R'_i)^2 = \{(R_i - R'_i) - (\bar{R} - \bar{R}')\}^2 = \{(R_i - \bar{R}) - (R'_i - \bar{R}')\}^2 \\ &= (R_i - \bar{R})^2 - 2(R_i - \bar{R})(R'_i - \bar{R}') + (R'_i - \bar{R}')^2 \end{aligned} \quad (5.1.7)$$

By reorganizing Equation (5.1.7), we get;

$$(R_i - \bar{R})(R'_i - \bar{R}') = \frac{1}{2} \left\{ - (R_i - R'_i)^2 + (R_i - \bar{R})^2 + (R'_i - \bar{R}')^2 \right\} \quad (5.1.8)$$

Because of the linearity of summation, from Equation (5.1.8), we get;

$$\sum_{i=1}^N (R_i - \bar{R})(R'_i - \bar{R}') = \frac{1}{2} \left\{ - \sum_{i=1}^N (R_i - R'_i)^2 + \underbrace{\sum_{i=1}^N (R_i - \bar{R})^2}_{(*1)} + \underbrace{\sum_{i=1}^N (R'_i - \bar{R}')^2}_{(*2)} \right\} \quad (5.1.9)$$

Now we focus on the part of Equation (5.1.9) and use Equations (5.1.3) and (5.1.5) we get the following;

$$\begin{aligned}
(*1) &= \sum_{i=1}^N (R_i - \bar{R})^2 = \sum_{i=1}^N R_i^2 - 2 \sum_{i=1}^N R_i \bar{R} + \sum_{r=1}^N \bar{R}^2 \\
&= \sum_{i=1}^N R_i^2 - 2(N\bar{R})\bar{R} + N\bar{R}^2 \\
&= \sum_{i=1}^N R_i^2 - N\bar{R}^2 = \frac{N(N-1)(2N-1)}{6} - N \cdot \frac{(N-1)^2}{4}
\end{aligned} \tag{5.1.10}$$

By using the same logic, we get:

$$(*2) = \sum_{T=1}^N (R'_i - \bar{R}')^2 = \sum_{i=1}^N R_i'^2 - N\bar{R}'^2 = \frac{N(N-1)(2N-1)}{6} - N \cdot \frac{(N-1)^2}{4} \tag{5.1.11}$$

Now, to make the calculation simpler, let's define  $\alpha$  as follows;

$$\alpha = \sum_{i=1}^N (R_i - \bar{R})^2 = \sum_{i=1}^N (R'_i - \bar{R}')^2 = \frac{N(N-1)(2N-1)}{6} - N \cdot \frac{(N-1)^2}{4} = \frac{N^3 - N}{12} \tag{5.1.12}$$

Consider Equation (5.1.9) again

$$\sum_{i=1}^N (R_i - \bar{R}) (R'_i - \bar{R}') = \frac{1}{2} \left\{ - \sum_{T=1}^N (R_i - R'_i)^2 + \alpha + \alpha \right\} \tag{5.1.13}$$

From Equations (5.1.6), (5.1.12), and (5.1.13), then we have

$$\begin{aligned}
r_s &= \frac{\frac{1}{2} \left\{ - \sum_{i=1}^N (R_i - R'_i)^2 + 2\alpha \right\}}{\sqrt{\alpha \cdot \alpha}} = \frac{-\frac{1}{2} \sum_{i=1}^N (R_i - R'_i)^2 + \alpha}{\alpha} \\
&= 1 - \frac{\frac{1}{2} \sum_{i=1}^N (R_i - R'_i)^2}{\alpha} = 1 - \frac{\frac{1}{2} \sum_{i=1}^N (R_i - R'_i)^2}{\frac{N^3 - N}{12}} \\
&= 1 - \frac{6 \sum_{i=1}^N (R_i - R'_i)^2}{N^3 - N} = 1 - \frac{6 \sum_{i=1}^N d_i^2}{N(N^2 - 1)}
\end{aligned}$$

Therefore, Equation (5.1.1) is proved by Equation (4.1.1).

## 5.2 Example I

The following calculation is made based on Equation (5.1.1).

```

rnk <- NULL
for (i in 1:20){rnk0 <- rank(data2[,i], ties.method = "average", na.last = "keep")
rnk <- as.data.frame(cbind(rnk, rnk0, row.names = NULL))
}# By using 'rank,' we can sort the data, 'data2,' in the ascending order.
# In the case where the multiple entries of initial response times are the same,
# the rank that will be assigned to each entry will be the averaged ranks,
# shown in 'ties.method = "average".'
# By using 'for,' we repeat the sorting process 20 times for each task. # The output is 'rnk.'
colnames(rnk) <- paste0("rankQ", 1:20)# We change the column names for 'rnk.'

```

```

fun2 <- function(i,j){
  Ri <- rnk[,i]
  Rj <- rnk[,j]
  N <- nrow(data2)
  rs <- 1- (6*sum((Ri-Rj)^2))/(N^3-N)
  return(rs)
}
# We define a new function, 'fun2,' which will return spearman correlations
# by inputting two column numbers, 'i' and 'j'.
result2 <- mapply(FUN=fun2, i=lis1, j=lis2, MoreArgs = NULL,
                  SIMPLIFY = TRUE, USE.NAMES = TRUE) %>% as.data.frame() %>% cbind(lis1, lis2)
# By using 'mapply,' we input the two lists, 'lis1' and 'lis2,' into 'fun2.'
# 'result2' stores the result of calculation for spearman correlations for
# all combinations of the 20 tasks and the two lists, 'lis1' and 'lis2.'
colnames(result2) <- c("rs", "lis1", "lis2") # We change the column names of result2.
rs0 <- dcast(result2, lis2 ~ lis1, value.var = "rs" )
rs0 <- rs0[,-1]
# By using 'dcast,' We can reorganize a long format data, 'result2,'
# and get a wide format data, 'rs0.'
colnames(rs0) <- paste0("Task", 1:20)
rownames(rs0) <- paste0("Task", 1:20)
# We change the column and row names of 'rs0'.
rs0<- round(rs0, 3)
# By using 'round,' we can change significant digits.
# We reduce the significant digits of each entry of 'rs0' to three decimals.
rs4 <- select(.data = rs0, -c(11:20))
rs5 <- select(.data = rs0, -c(1:10))
# By using 'select,' we separate the tables.
# By using 'kable,' we can make tables below.
kable(rs4, align = "c",
      caption = "***Table 5.2.1: Spearman Correlations of the initial response time for Task 1 to 20**",
      row.names = T, escape = FALSE, centering = T)

```

**Table 5.2.1: Spearman Correlations of the initial response time for Task 1 to 20**

	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10
Task1	1.000	0.539	0.500	0.487	0.486	0.470	0.474	0.464	0.467	0.453
Task2	0.539	1.000	0.594	0.567	0.565	0.559	0.545	0.542	0.542	0.542
Task3	0.500	0.594	1.000	0.606	0.597	0.577	0.591	0.542	0.560	0.542
Task4	0.487	0.567	0.606	1.000	0.606	0.597	0.588	0.578	0.578	0.564
Task5	0.486	0.565	0.597	0.606	1.000	0.605	0.622	0.609	0.581	0.570
Task6	0.470	0.559	0.577	0.597	0.605	1.000	0.605	0.594	0.593	0.569
Task7	0.474	0.545	0.591	0.588	0.622	0.605	1.000	0.610	0.594	0.581
Task8	0.464	0.542	0.542	0.578	0.609	0.594	0.610	1.000	0.617	0.611
Task9	0.467	0.542	0.560	0.578	0.581	0.593	0.594	0.617	1.000	0.615
Task10	0.453	0.542	0.542	0.564	0.570	0.569	0.581	0.611	0.615	1.000
Task11	0.470	0.526	0.512	0.520	0.539	0.517	0.552	0.555	0.540	0.534
Task12	0.460	0.503	0.520	0.526	0.545	0.547	0.574	0.573	0.569	0.566
Task13	0.431	0.486	0.499	0.513	0.514	0.545	0.548	0.549	0.555	0.548
Task14	0.399	0.492	0.490	0.502	0.512	0.508	0.535	0.524	0.550	0.537
Task15	0.400	0.460	0.472	0.482	0.504	0.509	0.521	0.510	0.538	0.522
Task16	0.397	0.466	0.475	0.476	0.504	0.513	0.525	0.505	0.538	0.510

	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Task10
Task17	0.382	0.442	0.464	0.477	0.491	0.492	0.512	0.520	0.523	0.524
Task18	0.391	0.445	0.466	0.465	0.492	0.505	0.525	0.512	0.527	0.517
Task19	0.369	0.443	0.440	0.463	0.491	0.507	0.507	0.512	0.515	0.490
Task20	0.374	0.448	0.463	0.461	0.482	0.492	0.507	0.508	0.523	0.497

```
kable(rs5)
```

	Task11	Task12	Task13	Task14	Task15	Task16	Task17	Task18	Task19	Task20
Task1	0.470	0.460	0.431	0.399	0.400	0.397	0.382	0.391	0.369	0.374
Task2	0.526	0.503	0.486	0.492	0.460	0.466	0.442	0.445	0.443	0.448
Task3	0.512	0.520	0.499	0.490	0.472	0.475	0.464	0.466	0.440	0.463
Task4	0.520	0.526	0.513	0.502	0.482	0.476	0.477	0.465	0.463	0.461
Task5	0.539	0.545	0.514	0.512	0.504	0.504	0.491	0.492	0.491	0.482
Task6	0.517	0.547	0.545	0.508	0.509	0.513	0.492	0.505	0.507	0.492
Task7	0.552	0.574	0.548	0.535	0.521	0.525	0.512	0.525	0.507	0.507
Task8	0.555	0.573	0.549	0.524	0.510	0.505	0.520	0.512	0.512	0.508
Task9	0.540	0.569	0.555	0.550	0.538	0.538	0.523	0.527	0.515	0.523
Task10	0.534	0.566	0.548	0.537	0.522	0.510	0.524	0.517	0.490	0.497
Task11	1.000	0.593	0.555	0.544	0.512	0.517	0.500	0.501	0.497	0.484
Task12	0.593	1.000	0.643	0.611	0.590	0.583	0.570	0.560	0.545	0.545
Task13	0.555	0.643	1.000	0.657	0.615	0.601	0.587	0.572	0.576	0.571
Task14	0.544	0.611	0.657	1.000	0.626	0.617	0.596	0.593	0.583	0.590
Task15	0.512	0.590	0.615	0.626	1.000	0.637	0.605	0.600	0.574	0.570
Task16	0.517	0.583	0.601	0.617	0.637	1.000	0.626	0.614	0.594	0.588
Task17	0.500	0.570	0.587	0.596	0.605	0.626	1.000	0.640	0.602	0.593
Task18	0.501	0.560	0.572	0.593	0.600	0.614	0.640	1.000	0.624	0.594
Task19	0.497	0.545	0.576	0.583	0.574	0.594	0.602	0.624	1.000	0.613
Task20	0.484	0.545	0.571	0.590	0.570	0.588	0.593	0.594	0.613	1.000

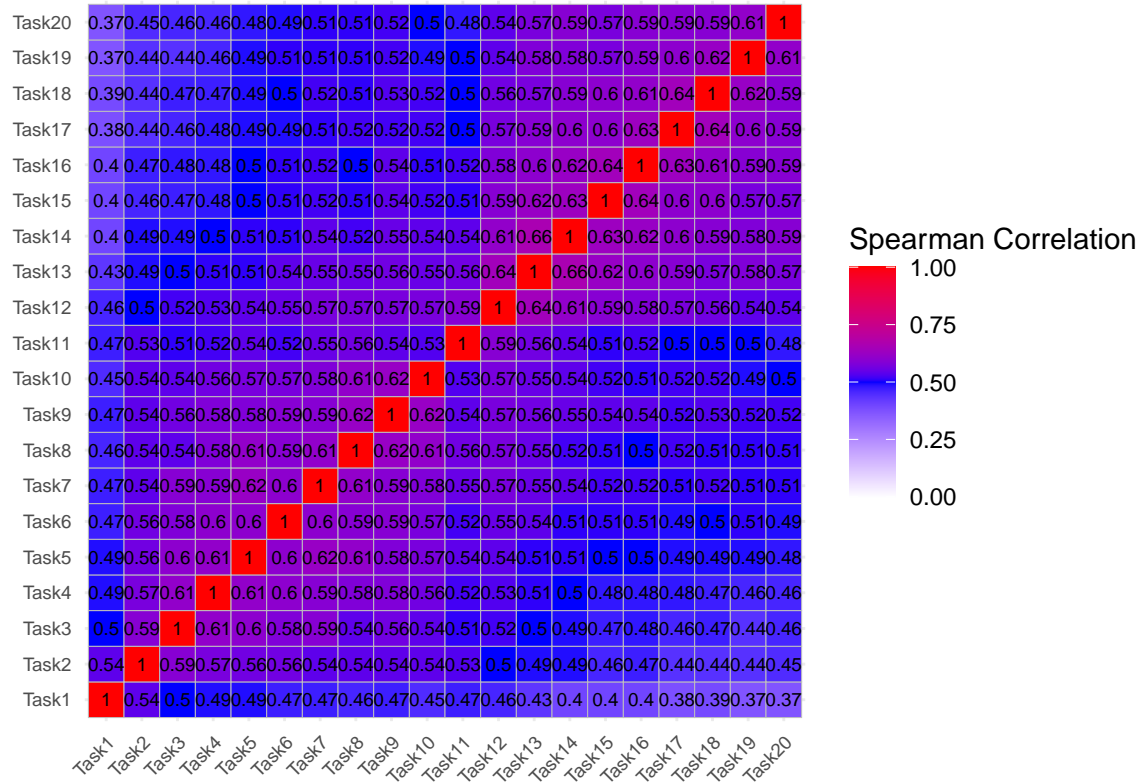
In addition to ‘ggplot’ and ‘corrplot,’ ‘ggcorrplot’ is one of the options to visualize correlation matrices as a heat-map. ‘ggplot’ and ‘ggcorrplot’ have comparatively similar looks so if you simply want to add coefficients to ‘ggplot,’ it is better to use ‘ggcorrplot’ than ‘corrplot.’

By comparing Figure 4.2.2 and Figure 5.2.2, we notice Spearman’s rank correlation coefficients are bigger than Pearson correlation coefficients even though they are calculated for the same data. This is because, as mentioned before, Pearson correlations only capture the linear relations while Spearman’s correlation could capture the monotonic relations. When there exist a monotonic relation, but it is not a linear relation, we have  $-1 \leq r_p \leq r_s \leq 1$ .

In this sense, by comparing Figure 4.2.2 and Figure 5.2.2, we conclude that there exist non-linear monotonic relations for some of the initial response time to 20 tasks.

```
ggcorrplot(rs0, hc.order = F, lab = TRUE, tl.cex = 7, lab_size = 2.5, ) +
  scale_fill_gradient2(low = "white", high = "red", mid = "blue",
    midpoint = 0.5, limit = c(0,1),
    name = "Spearman Correlation", space = "Lab") +
  labs(x = "Task Number", y = "Task Number",
    title = "Figure 5.2.2: Spearman Correlation of the initial response time")
```

Figure 5.2.2: Spearman Correlation of the initial response time



```
# By using 'ggcorrplot,' we can make a heat-map with indicating coefficients.
# We can input list into 'ggcorrplot.'
# Since we input a wide format data.frame, 'rs0,' we do not need to indicate axes.
# By 'scale_fill_gradient2,' we can change the format of the figure.
# By 'labs,' we can add labels to x-axis, y-axis, and the title of the figure.
```

## 6 Reference

Jakubczyk, M., Craig, B. M., Barra, M., Groothuis-Oudshoorn, C. G. M., Hartman, J. D., Huynh, E., Ramos-Goñi, J. M., Stolk, E. A., & Rand, K. (2017). Choice defines value: A predictive modeling competition in Health Preference Research. *Value in Health*, 21(2), 229–238. <https://doi.org/10.1016/j.jval.2017.09.016>

## 7 How to cite this entry

Okubo, S. (yyyy, month dd). Correlations between initial response times. *R4HPR*. <https://r4hpr.org/visor/?e=correlations-between-initial-response-times%e3%80%80>