Satistical analysis of the Stigmer data

Contents

Packages needed:

```
require("tidyverse")
```

Vocabulary used:

- there are three **rules** : R0, R1 and R2,
- a session is constituted of 15 **rounds**,
- Each round is constituted of 5 **steps**,
- The **gain per step** is equal to 0, 15 (or 50), 99,
- The **gain per round** is the sum of the 5 gains obtained during a round,
- The **final gain** is the sum of the gains obtained by a player during a session,
- A **cell** is one of the 9 cells of the grid,
- Under R1, a player let a **coin** at each cell visited,
- Under R2, a player has the choice to let or not a coin at the visited cell.

1 Importation of the data

The function *import_files()* leads us to import the data depending on the intermediate value in the game (15 or 50).

```
import_files <- function(game) {
  # imput: game is an integer with value 15 or 50
  # this function imports all the data included in the directory data/Stigmer-GTgame
  # for the three rules (R0, R1, R2)
  # output: a data frame. Each row contains the results (cells + token played) at each of the 5 step
  # We also have information on the rule, the session, the player.

  stopifnot(game %in% c(15, 50))
  stigmer <- paste0("Stigmer-GT", game)
  fic_final <- NULL
  file_with_problem <- NULL

  for (rule in c(0, 1, 2)){
    cat("Rule ", rule, "\n")
    session_list <- sapply(strsplit(dir(paste0("data/", stigmer, "/Regle ", rule, "/OUT/")),
                                    "Session "),
                           function(x) x[2])
```

```
    for (session in session_list) {
      cat("Session ", session, "\n")
      directory <- paste0("data/", stigmer, "/Regle ", rule, "/OUT/Session ", session, "/")

      list_file <- dir(directory)

      for (k in list_file) {
        fic <- read.csv2(paste0(directory, k))
        tour <- substr(k, 9, 10) # Tour
        fic <- fic[-1, ]
        fic$case <- paste(fic$x, fic$y, sep = "_")
        # test nombre de lignes
        if(nrow(fic) != 25){
          cat("File : ", k, "has been imported but it has only",  nrow(fic), "observations \n")
          file_with_problem <- c(file_with_problem, paste0(directory, k))
        }
        # for the gain
        gain <- fic[, c("joueur", "tour", "points.gagnés")] %>% spread(tour, points.gagnés)
        names(gain)[2:6] <- paste("gain", 1:5, sep = "_")
        gain$gain <- apply(gain[, 2:6], 1, sum)
        # for the case
        cell <- fic[, c("joueur", "tour", "case")] %>% spread(tour, case)
        names(cell)[2:6] <- paste("cell", 1:5, sep = "_")
        # for the action
        action <- fic[, c("joueur", "tour", "pièces.misées")] %>% spread(tour, pièces.misées)
        names(action)[2:6] <- paste("action", 1:5, sep = "_")
        # merge the data
        fic <- merge(merge(gain, cell), action)
        # rename the variable
        fic <- dplyr::rename(fic, player = joueur)
        # add variables
        fic$rule <- paste("R", rule, sep = "")
        fic$session <- paste0("session_", session)
        fic$round <- paste0("T", tour, sep = "")
        fic$file <- paste(stigmer, session, paste0(directory, k), sep = "_")
        fic_final <- rbind(fic_final, fic)
      }
    }
  }
  return(fic_final)
}
```

To import the data, we call the previous function:

```
file_15 <- import_files(15)
file_50 <- import_files(50)
```

1.1 Detection of anomalies

These anomalies have been detected in the version of the data given on March the 5th 2018

1.1.1 Stigmer 15

- In: Rule 1, session 03, in round "7B", a player is called 4B instead of B4. We correct this anomaly:

```
file_15[file_15$player == "4B", "player"] <- "B4"
```

- In: Rule 2, session 01, we found two rounds with name "14A" : we kept only one round (the one with the date the earlier).
- In: Rule 2, session 03, there is one missing row in round "11A". A player did not play at step 1. We impute this missing value by considering that he had a higher probability to get a 0. We correct this:

```
file_15[is.na(file_15$gain_1), c("gain_1", "action_1", "gain")] <-
    c(0, 0, 114)
file_15[is.na(file_15$gain_1), "cell_1"] <- "1_1"
```

- In Rule 1, players should have let a coin at each step. We notice that this is not necessarly the case :
  - player A2 in session 01, round "1A", "3A", "5A", "12A", "13A",
  - player B1 in session 02, round "6B", "7B", "8B", "12B",
  - player B2 in session 01, round "6B",
  - player A2 in session 02, round "11A",
  - player A1 in session 03, round "1A",
  - player B4 in session 03, round "6B".

Remark: no corrections ave been applied because in Rule 1, we implicitely suppose that all players have indicated their choice.

- We now identify the players who did not play value 99 although they knew where it was located. For doing this, we create a function :

```
detec_bad_player <- function (fic){
  # this function prints the players who behave anormaly
  # input: the name of the file to deal with
  for (k in 1:3) {
    fic_rule <- filter(fic, rule == paste0("R", k - 1))
    ind_bad <- NULL
    for (j in 1:4) {
      col_ind <- which(colnames(fic_rule) == paste0("gain_", j))
      ind_99 <- which(fic_rule[, col_ind] == 99)
      ind_bad <- c(ind_bad, ind_99[which(apply(fic_rule[ind_99, col_ind:6], 1,
                                               function(x) any(x != 99)))])
    }
    ind_bad <- unique(ind_bad)
    cat("** Bad players in", paste0("R", k - 1), "\n")
    ind_ord <- order(fic_rule$rule[ind_bad], session = fic_rule$session[ind_bad],
                     player = fic_rule$player[ind_bad], round = fic_rule$round[ind_bad])
    unique_player_session <- unique(fic_rule[ind_bad[ind_ord], c("session", "player")])
    for (i in 1:nrow(unique_player_session)){
      cat("In", unique_player_session$session[i], ", player",
          as.character(unique_player_session$player[i]), "has played: \n")
      ind_sub_base <- fic_rule[ind_bad[ind_ord], "session"] == unique_player_session$session[i] &
        fic_rule[ind_bad[ind_ord], "player"] == unique_player_session$player[i]
      print(fic_rule[ind_bad[ind_ord][ind_sub_base],
                     c("round", "gain_1", "gain_2", "gain_3", "gain_4", "gain_5")])
    }
    cat("\n")
```

```
  }
}
```

We print the bad players :

```
detec_bad_player(file_15)
```

```
## ** Bad players in R0
## In session_03 , player A5 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 155   T01      0     99     15      0      0
## In session_03 , player B3 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 158   T01      0      0     15     99     15
## In session_03 , player B4 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 179   T03      0     15     15     99      0
## 209   T06     15     99     15      0      0
## 229   T08     15      0     99      0     15
## 249   T10      0     15     99      0      0
## 259   T11      0     15     15     99      0
## 279   T13     99     15      0      0      0
## In session_03 , player B5 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 230   T08     99     99     99     99      0
##
## ** Bad players in R1
## In session_01 , player A3 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 3     T01      0      0     99     15      0
## 13    T02      0      0     99      0     15
## 23    T03      0     15     15     99     15
## 33    T04     99     15      0      0      0
## 53    T06     15      0     99      0      0
## 63    T07     15     99      0      0      0
## 73    T08     15     15      0     99      0
## 83    T09      0      0     99     15      0
## 103   T11      0      0     99     15     15
## 113   T12     15     99     15      0     15
## 133   T14     15      0     99      0      0
## 143   T15      0      0     15     99     15
## In session_03 , player B4 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 319   T02      0     99      0      0     15
## 366   T07      0     15     15     99      0
## 389   T09      0      0     99     15      0
## 429   T13      0     15     99      0     15
## 449   T15     99     15      0     15      0
##
## ** Bad players in R2
## In session_01 , player A3 has played:
##   round gain_1 gain_2 gain_3 gain_4 gain_5
## 3   T01     15      0     99      0      0
## In session_03 , player B4 has played:
```

```
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 479   T03      0     99      0      0     15
## 489   T04      0     15      0     99      0
## 499   T05      0     99      0     15     15
## 559   T11     99      0      0     15      0
## 569   T12      0      0     99     15      0
## 599   T15     15     99      0      0     15
```

Remark: we do not apply any corrections for these individuals and keep them in the analysis.

1.1.2 Stigmer 50

- In : Rule 0 / session 01 / round "3B", player B4 did not play at all during this game. Correction done: none.
- In Rule 1, players should have indicated all their choices (value 1). We notice that this not necessarly the case :
  - player B2 in session 01, round "1B",
  - player A3 in session 01, round "3A", "5A",
  - player A4 in session 01, round "1A",
  - player A3 in session 01, round "2A", "3A", "4A".

Remark: no corrections have been applied because in Rule 1, we implicitely suppose that all players have indicated their choice.

- We now identify the players who did not play value 99 although they know where it is located :

```
detec_bad_player(file_50)
```

```
## ** Bad players in R0
## In session_01 , player A1 has played:
##    round gain_1 gain_2 gain_3 gain_4 gain_5
## 30   T04     99      0     50     99     99
## 40   T05     50      0     99      0      0
## 60   T07      0      0     99      0      0
## In session_02 , player A2 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 171   T03      0      0     99     50     50
## 181   T04     50     50     99      0      0
## 201   T06     50      0     99     50     50
## 261   T12     99     50      0      0      0
## 271   T13      0      0      0     99     50
## 291   T15      0     50     99      0     50
## In session_02 , player B3 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 187   T04     50      0     99     50     50
## 207   T06      0     50     99      0     50
## 227   T08     99      0     50     50     50
## 237   T09     99     99     50     50     50
##
## ** Bad players in R1
## In session_02 , player B3 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 165   T02      0     99      0     99     99
```

```
## 218   T07      0     99     99     99     50
##
## ** Bad players in R2
## In session_01 , player B2 has played:
##     round gain_1 gain_2 gain_3 gain_4 gain_5
## 127   T13     99     50     99     99     99
```

Remark: we do not apply any corrections for these individuals and keep them in the analysis.

2 Statistical analysis of Stigmer 15

2.1 Statistical distribution of the gain per step

The idea here is to represent the empirical probability to get one of the values 0, 15 (or 50), 99 at each of the 5 steps, depending on the rule (0, 1 or 2).

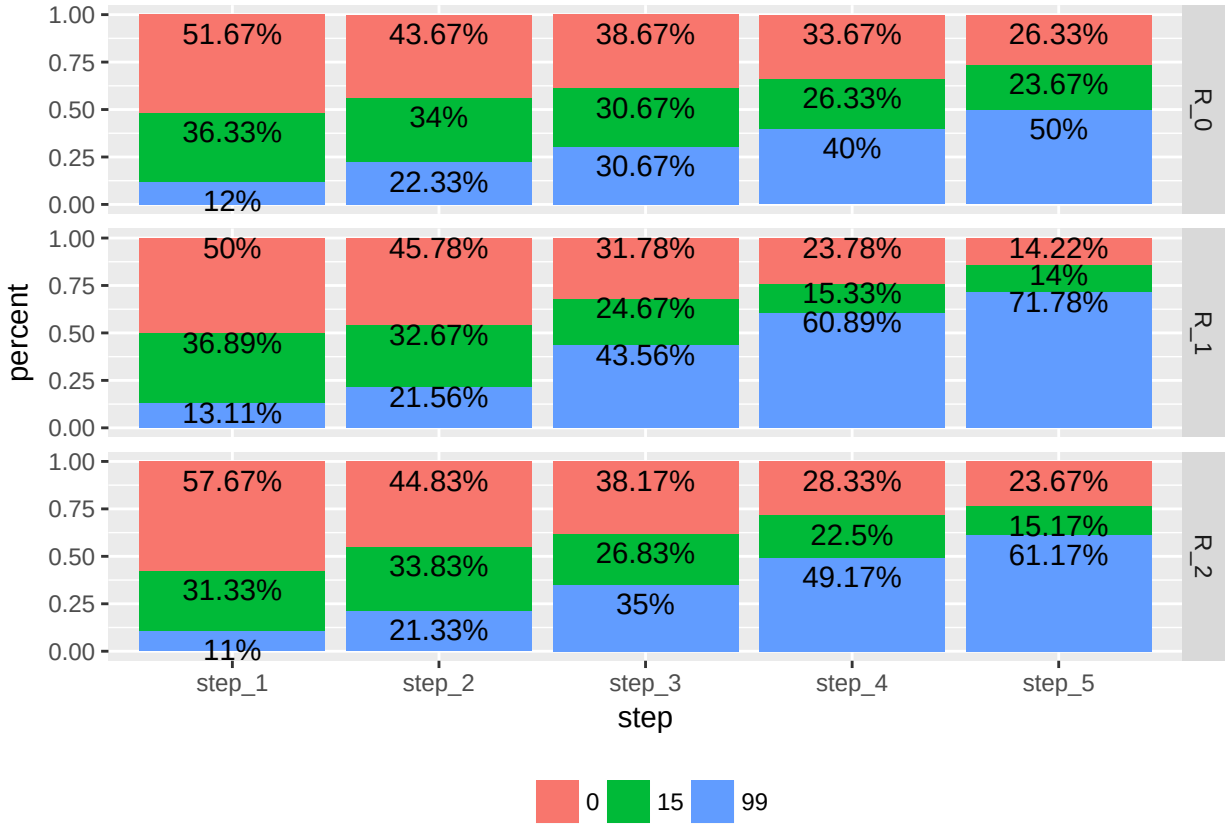
2.1.1 Representation

We create a function which plots the statistical distribution.

```
distib_per_step <- function (fic, game) {
  # compute the distribution
  # fic: the name of the file
  # game: the value of the intermediate value (15 or 50)
  dist_gain <- list(R_0 = NULL, R_1 = NULL, R_2 = NULL)
  for(k in 1:3){
    dist_gain[[k]] <- matrix(0, 3, 5)
    row.names(dist_gain[[k]]) <- paste0("game", c(0, game, 99))
    colnames(dist_gain[[k]]) <- paste0("step_", 1:5)
    fic_rule <- filter(fic, rule == paste0("R", k - 1))
    n <- nrow(fic_rule )
    for (j in 1:5){
      dist_gain[[k]][, j] <- table(fic_rule[, which(colnames(fic_rule) == paste0("gain_", j)) ])
      }
    dist_gain[[k]] <- dist_gain[[k]]/n
  }
  # We transform the data to facilitate the graphical representation
  dist_gain_R0 <- as.data.frame(dist_gain$R_0) %>% gather(step, percent)
  dist_gain_R0$gain <- rep(c("0", game, "99"), 5)
  dist_gain_R0 <- plyr::ddply(dist_gain_R0, plyr::.(step),
                              transform, pos = percent + (0.9 - cumsum(percent)))
  dist_gain_R1 <- as.data.frame(dist_gain$R_1) %>% gather(step, percent)
  dist_gain_R1$gain <- rep(c("0", game, "99"), 5)
  dist_gain_R1 <- plyr::ddply(dist_gain_R1, plyr::.(step),
                              transform, pos = percent + (0.95 - cumsum(percent)))
  dist_gain_R2 <- as.data.frame(dist_gain$R_2) %>% gather(step, percent)
  dist_gain_R2$gain <- rep(c("0", game, "99"), 5)
  dist_gain_R2 <- plyr::ddply(dist_gain_R2, plyr::.(step),
                              transform, pos = percent + (0.9 - cumsum(percent)))
  dist_gain_plot <- rbind(dist_gain_R0, dist_gain_R1, dist_gain_R2)
  dist_gain_plot$rule <- rep(c("R_0", "R_1", "R_2"), each = 15)
  # we represent the statistical distribution:
```

```
  p4 <- ggplot() +
    geom_bar(aes(y = percent, x = step, fill = gain),
             data = dist_gain_plot, stat="identity") +
    geom_text(data = dist_gain_plot, aes(x = step, y = pos,
                                         label = paste0(round(percent * 100, 2) ,"%")),
              size = 4) +
    theme(legend.position = "bottom", legend.direction = "horizontal",
          legend.title = element_blank()) +
    facet_grid(rule ~ .)
  print(p4)
}
```

```
distib_per_step(fic = file_15, game = 15)
```



Remark: at step 1 and 2, the distributions seem to be the same under R0, R1 and R2. From step 3, we remark some differences between the rules. From example, at step 3 under R0, the probablity to find 99 or 15 is the same and slightly smaller than the probability to find 0. On contrary, under R1, the probability to find 99 is higher than the probabilities to get 0 neither 15. The distribution under R2 seem to be a mixture of the distributions under R0 and R2.

2.2 Analysis under R0

2.2.1 Probability to get 0, 15, 99 under R0: comparaison between empirical and expected distribution

In R0, to maximize his profit, a player should explore a new cell at each new step until he finds the value 99 (this could be prooved by using theoretical probability). Thus, we can simulate a high number of times this

kind of behaviour, such that we have the expected distribution.

```
simu_distrib_R0 <- function(game) {
  # game : the value of the intermediate value : 15 or 50
  res <- numeric(5)
  nb_choice <- 9
  res[1] <- c(0, game, 99)[which(rmultinom(1, size = 1,
                                           prob = c(5/nb_choice,
                                                    3/nb_choice,
                                                    1/nb_choice)) == 1)]
  joue_0 <- (res[1] == 0)
  joue_game <- (res[1] == game)
  tour <- 2
  nb_choice <- 8

  while (tour <= 5) {
    if (res[tour - 1] == 99) {
      res[tour:5] <- 99
      break
      } else {
        if (res[tour - 1] == 50) {
          res[tour:5] <- 50
          break
          } else {
            res[tour] <- c(0, game, 99)[which(rmultinom(1, size = 1,
                                                        prob = c((5 - joue_0)/nb_choice,
                                                                 (3 - joue_game)/nb_choice,
                                                                 1/nb_choice)) == 1)]
            joue_0 <- joue_0 + (res[tour] == 0)
            joue_game <- joue_game + (res[tour] == game)
            nb_choice <- nb_choice - 1
            tour <- tour + 1
          }
      }
    }
  return(res)
}
```

We simulate the behaviours of 100000 players :

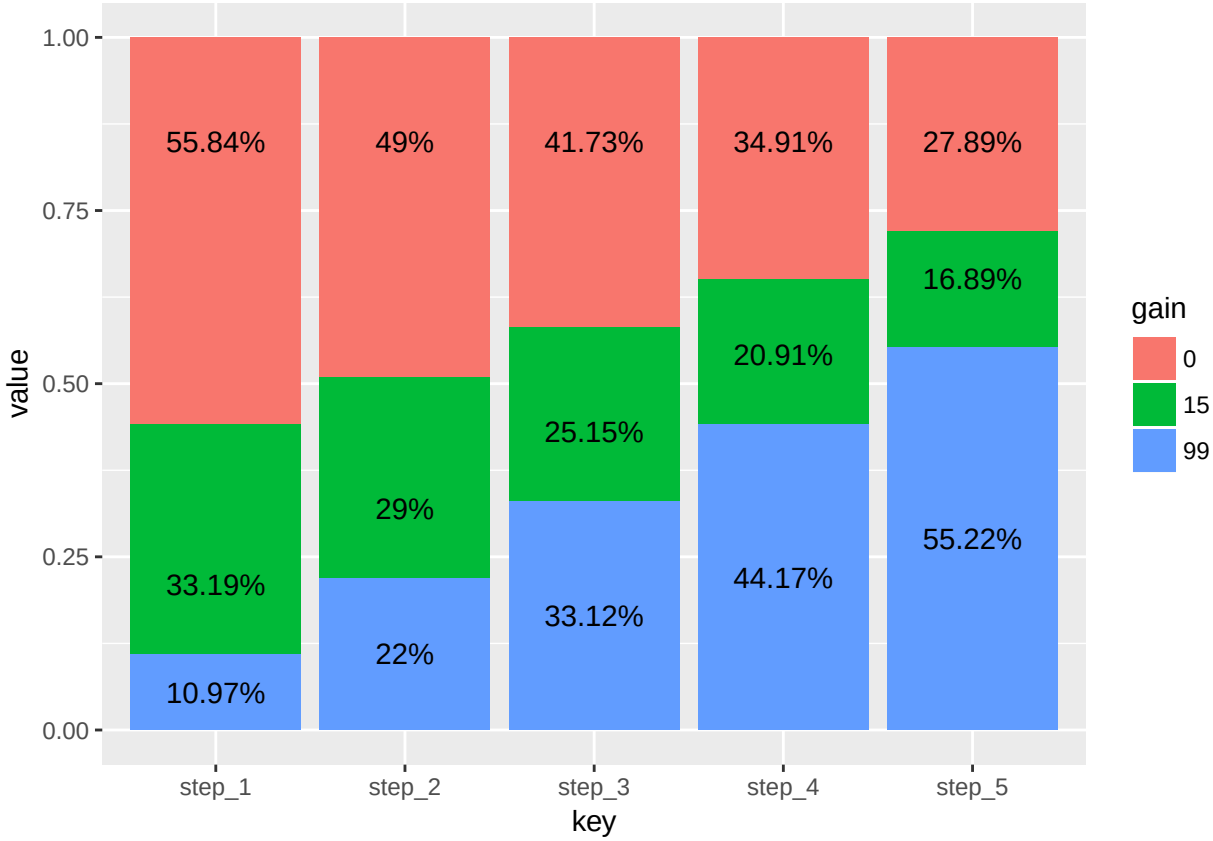
```
B <- 100000
distrib_15 <- replicate(B, simu_distrib_R0(15))
```

We obtain this distribution :

```
tab_15 <- apply(distrib_15, 1, table)/B
colnames(tab_15) <- paste0("step_", 1:5)
gather_tab_15 <- gather(as.data.frame(tab_15))
gather_tab_15$gain <- factor(rep(c(0, 15, 99), 5))
gather_tab_15$pos <- NULL
gather_tab_15$pos[gather_tab_15$gain == "0"] <- 0.85
gather_tab_15$pos[gather_tab_15$gain == "99"] <- gather_tab_15$value[gather_tab_15$gain == "99"]/2
gather_tab_15$pos[gather_tab_15$gain == "15"] <- 0.1 + gather_tab_15$value[gather_tab_15$gain == "99"]
```

We finally plot the theoretical distribution:

```
ggplot(gather_tab_15, aes(y = value, x = key, fill = gain)) +
  geom_bar(stat = "identity") +
  geom_text(aes(x = key, y = pos, label = paste0(round(value * 100, 2) ,"%")),
            size = 4)
```



We compare below the empirical distribution with the theoretical one obtained previously. We use a χ^2 test which consists in comparing at each step the empirical distribution of 0, 15, 99 to the theoretical one computed previously. It seems that the more we progress in the experience, the less the players seem to behave as players who optimize their profit. This is probably due to the fact that some players prefer to conserve their results by playing the value 15 (when they know where it is located) rather than continuing to explore, specially at the end of a round.

Interpretation of the χ^2 test: the null hypothesis is "Distribution of empirical and theoretical are identical". When the p-value is lower than 0.05, we usually reject the null hypothesis. If the p-value is upper than 0.05, we cannot reject it. With our data, with a level of 5%, the distributions (emprirical VS theoretical) are not the same only at step 5.

```
file_15_R0 <- file_15[file_15$rule == "R0", ]
```

```
chisq.test(table(as.factor(file_15_R0[, "gain_1"])),
           p = tab_15[, 1])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R0[, "gain_1"]))
## X-squared = 2.12, df = 2, p-value = 0.3465
```

```
chisq.test(table(as.factor(file_15_R0[, "gain_2"])),
           p = tab_15[, 2])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R0[, "gain_2"]))
## X-squared = 4.3435, df = 2, p-value = 0.114
```

```
chisq.test(table(as.factor(file_15_R0[, "gain_3"])),
           p = tab_15[, 3])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R0[, "gain_3"]))
## X-squared = 4.8557, df = 2, p-value = 0.08823
```

```
chisq.test(table(as.factor(file_15_R0[, "gain_4"])),
           p = tab_15[, 4])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R0[, "gain_4"]))
## X-squared = 5.5316, df = 2, p-value = 0.06293
```

```
chisq.test(table(as.factor(file_15_R0[, "gain_5"])),
           p = tab_15[, 5])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R0[, "gain_5"]))
## X-squared = 9.9017, df = 2, p-value = 0.007077
```

2.2.2 Gain per round under R0: comparaison between empirical and expected distribution

Thanks to the previous simulation, we can obtain the statistical distibution of the gain per round under R0 when players adopt a optimal behaviour.

```
final_gain_15 <- apply(distrib_15, 2, sum)
```

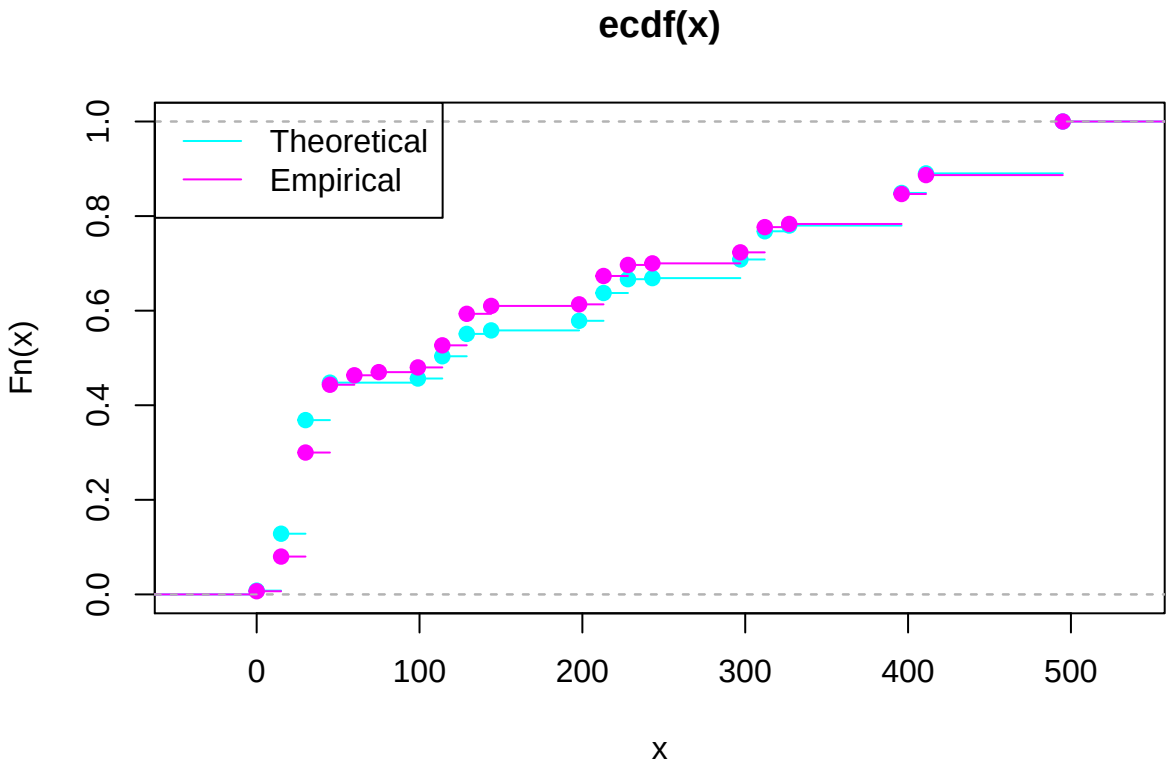
Example of interpretation of the empirical cumulative distribution function :

- the probability to obtain a gain per round lower or equal to 30 is 36.41%.
- 49.925% of the population obtained a value lower or equal to 114. On contrary, $100 - 49.925 = 50.075\%$ obtained a gain per round strictly larger than 114.

```
cumsum(prop.table(table(final_gain_15)))
```

```
##       0      15      30      45      99     114     129     144     198
## 0.00801 0.12835 0.36854 0.44777 0.45651 0.50358 0.55088 0.55825 0.57854
##     213     228     243     297     312     327     396     411     495
## 0.63736 0.66656 0.66881 0.70825 0.76787 0.77996 0.84890 0.89027 1.00000
```

```
plot.ecdf(final_gain_15, col = "cyan")
plot.ecdf(file_15_R0[, "gain"], col = "magenta", add = T)
legend("topleft", legend = c("Theoretical", "Empirical"),
       lty = 1, col = c("cyan", "magenta"))
```



The plot of the empirical cumulative distribution function ("theoretical" versus "empirical") does not show a big difference between the two distributions since the two curves (cyan vs magenta) seem close.

We use the Kolmogorov-Smirnov test to verify that the two distributions (empirical and theoritical) are extracted from the same distribution or not. The null hypothesis which is "the two distributions are identical" canno't be rejected here. In other term, the gain per round obtained by players could be the ones obtained by players who optimize their profit.

```
ks.test(x = final_gain_15,
        y = file_15_R0[, "gain"],
        exact = F)
```

```
## Warning in ks.test(x = final_gain_15, y = file_15_R0[, "gain"], exact = F):
## p-value will be approximate in the presence of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  final_gain_15 and file_15_R0[, "gain"]
## D = 0.06854, p-value = 0.1204
## alternative hypothesis: two-sided
```

Remark: note that the Kolmogorov-Smirnov test is supposed to be used on continous variable which is note the case here (the gain per round is indeed discrete). a solution could be to use instead a χ^2 test. However, we can not use it for the following reason : in the optimal situation (theoretical distribution), players might not be able to obtain a gain per round equal to 60 or 75 (a player is exploring new cases unless he founds 99, so he can only find the value 15 one, two or three times). Thus, the probability to obtain these values 60 or

75 are equal to 0, although these situations can occur in the experimental context (players who play several times the same cell containing 15). Besides, the χ^2 test can be seen as the sum of the distances between theoretical and empirical values divided by the theoretical probability: thus, this is an issue when probability equals to 0.

2.2.3 Behaviours among the players under R0

We show previously that players seem to behave as players who maximise their profit (exploring until the value 99 is found). However, we are interested to check that all players behave like that and if not, could we notice differences in the final gain. For each player, we compute the final gain after the 15 rounds and compute the percentage of time they have chosen to continue to explore new cells until they find 99.

We identify players who have the opportunity to explore at a given step and do it (or not). For doing that, we create a function that will be also used in next sections:

```
exploring <- function (fic) {
  # function which creates variables to identify players who
  # explore or not in a situation where they have the possibility to do it
  fic$explore_1 <- ifelse((fic$gain_1 == 99), NA,
                          ifelse((fic$gain_1 != 99) & (fic$cell_2 != fic$cell_1),
                                 "yes", "no"))
  fic$explore_2 <- ifelse((fic$gain_2 == 99), NA,
                          ifelse((fic$gain_2 != 99) & (fic$cell_3 != fic$cell_2) &
                                   (fic$cell_3 != fic$cell_1),
                                 "yes", "no"))
  fic$explore_3 <- ifelse((fic$gain_3 == 99), NA,
                          ifelse((fic$gain_3 != 99) & (fic$cell_4 != fic$cell_3) &
                                   (fic$cell_4 != fic$cell_2) &
                                   (fic$cell_4 != fic$cell_1),
                                 "yes", "no"))
  fic$explore_4 <- ifelse((fic$gain_4 == 99), NA,
                                 ifelse((fic$gain_4 != 99) & (fic$cell_5 != fic$cell_4) &
                                          (fic$cell_5 != fic$cell_3) &
                                          (fic$cell_5 != fic$cell_2) &
                                          (fic$cell_5 != fic$cell_1),
                                        "yes", "no"))

  # We count the number of times a player has explored new cells during a round :
  fic$explore_yes <- apply(fic[, c("explore_1", "explore_2", "explore_3", "explore_4")],
                           1, function(x) length(which(x == "yes")))
  fic$explore_no <- apply(fic[, c("explore_1", "explore_2", "explore_3", "explore_4")],
                          1, function(x) length(which(x == "no")))
  fic$explore_yes_no <- fic$explore_yes + fic$explore_no

  return(fic)
}
```

We use the function like this :

```
file_15_R0_explore <- exploring(file_15_R0)
```

2.2.3.1 Boxplots of the gain per round obtained depending on the fact that a player continue to explore or not

In this section, we propose to compare the gain per round obtained depending on the fact that a player had the opportunity to explore new cells.

First, we tidy the data:

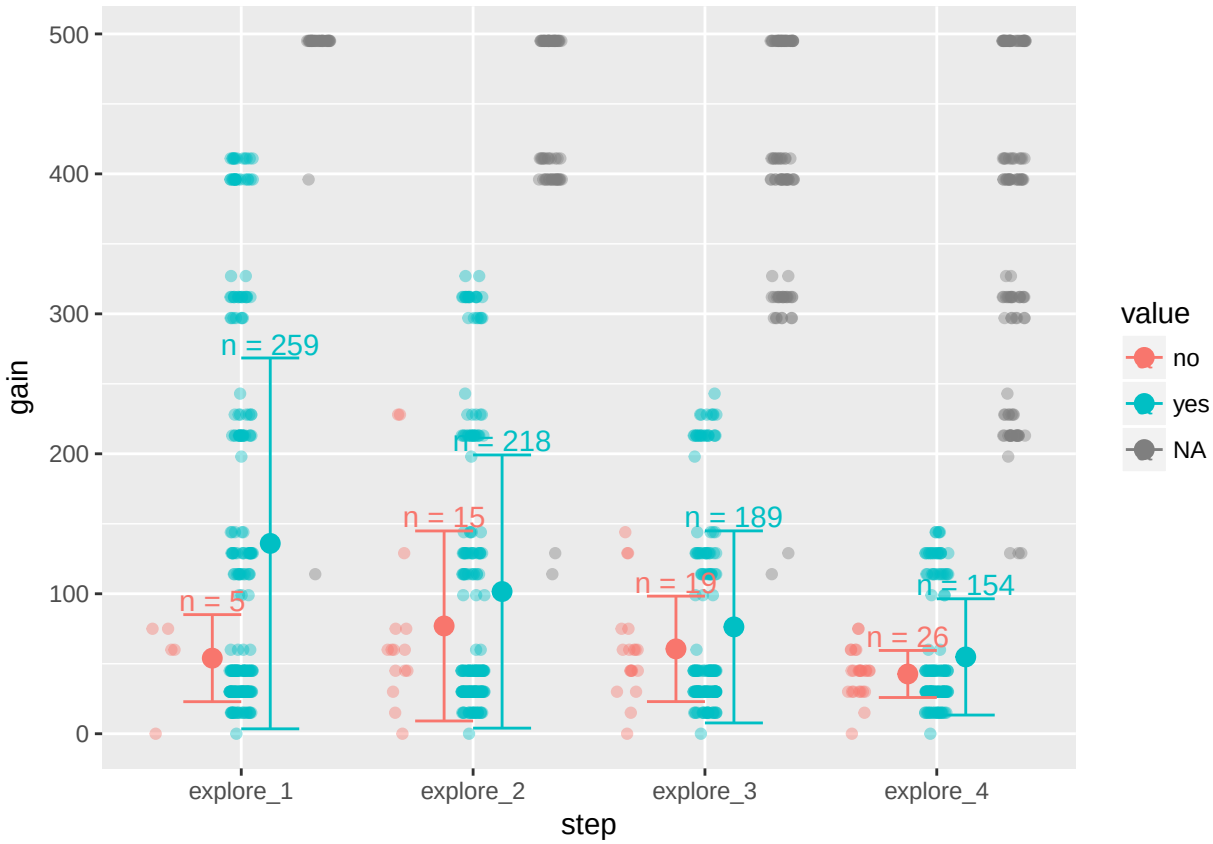
```
gather_file_15 <- file_15_R0_explore %>%
  select(explore_1, explore_2, explore_3, explore_4, gain) %>%
  gather(gain)
names(gather_file_15)[2] <- "step"
```

We compute the mean and standard deviation obtained at each step and depending on the fact that a player explored new cells or not :

```
stat_gather_file_15 <- merge(aggregate(gather_file_15$gain,
                                       by = list(step = gather_file_15$step,
                                                 value = gather_file_15$value), mean),
                             merge(aggregate(gather_file_15$gain,
                                             by = list(step = gather_file_15$step,
                                                       value = gather_file_15$value), sd),
                                   aggregate(gather_file_15$gain,
                                             by = list(step = gather_file_15$step,
                                                       value = gather_file_15$value), length),
                                   by = c("step", "value")),
                             by = c("step", "value"))
names(stat_gather_file_15)[3:5] <- c("gain", "sd", "length")
```

We plot in y-axis the gain per round and in x-axis the steps. We represent in blue (resp. in red) the gain per round obtained when a player explored new cells (resp. when he did not explore) knowing that the player had the opportunity to do it. We plot the gain per round in grey obtained by people who did not have the opportunity to explore (that means that they know where the cell 99 is located):

```
ggplot(gather_file_15, aes(x = step, y = gain, colour = value)) +
  geom_jitter(alpha = I(2/5),
              position = position_jitterdodge(jitter.width = 0.1, dodge.width = 1)) +
  geom_errorbar(data = stat_gather_file_15, aes(ymin = gain - sd,
                                                ymax = gain + sd, colour = value),
                width = 0.5,
                position = position_dodge(width = 0.5)) +
  geom_point(data = stat_gather_file_15, aes(y = gain, x = step, colour = value),
             position = position_dodge(width = 0.5), size = 3) +
  geom_text(data = stat_gather_file_15, aes(y = 10 + gain + sd, label = paste0("n = " , length)),
            position = position_dodge(width = 0.5))
```



This plot confirms that players who explore seem to obtain "in average" a higher gain per round than people who does not explore. It is also interesting to notice that the standard error of the red part is always including in the standard error of the blue part.

To test the hypothesis of equality of means at each step, we use a non parametric test called "Kruskal-Wallis Rank Sum Test". It shows us that the differences of the means between players who explore and players who do not explore are non significant at any step.

```
(kruskal.test(gather_file_15$gain[gather_file_15$step == "explore_1"] ~
                           factor(gather_file_15$value[gather_file_15$step == "explore_1"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15$gain[gather_file_15$step == "explore_1"] by factor(gather_file_15$value[gather
## Kruskal-Wallis chi-squared = 0.40938, df = 1, p-value = 0.5223
```

```
(kruskal.test(gather_file_15$gain[gather_file_15$step == "explore_2"] ~
                           factor(gather_file_15$value[gather_file_15$step == "explore_2"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15$gain[gather_file_15$step == "explore_2"] by factor(gather_file_15$value[gather
## Kruskal-Wallis chi-squared = 0.049873, df = 1, p-value = 0.8233
```

```
(kruskal.test(gather_file_15$gain[gather_file_15$step == "explore_3"] ~
                           factor(gather_file_15$value[gather_file_15$step == "explore_3"])))
```

```
##
```

```
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15$gain[gather_file_15$step == "explore_3"] by factor(gather_file_15$value[gather
## Kruskal-Wallis chi-squared = 0.54774, df = 1, p-value = 0.4592
```

```
(kruskal.test(gather_file_15$gain[gather_file_15$step == "explore_4"] ~
                           factor(gather_file_15$value[gather_file_15$step == "explore_4"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15$gain[gather_file_15$step == "explore_4"] by factor(gather_file_15$value[gather
## Kruskal-Wallis chi-squared = 0.27042, df = 1, p-value = 0.603
```

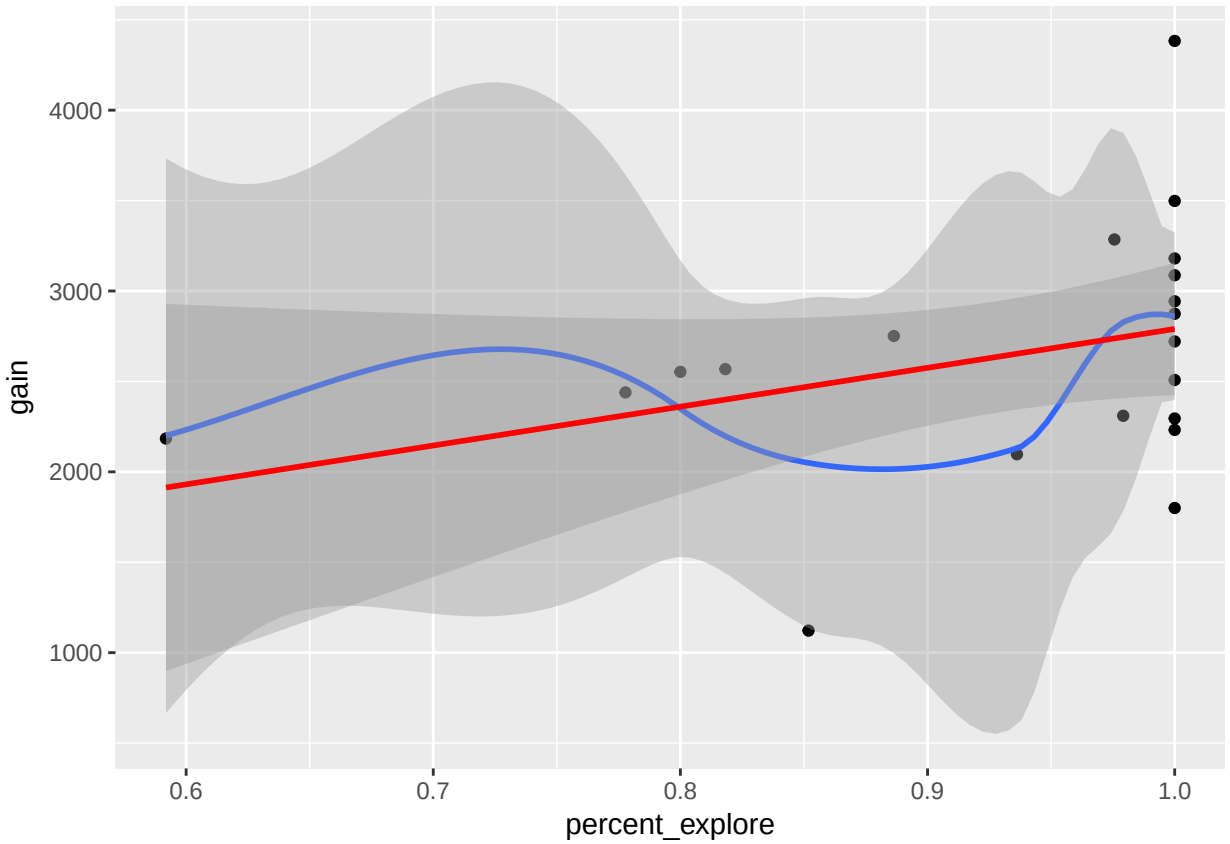
2.2.3.2 The effect of exploring (or not) on the final gain

We aggregate the results for all the rounds and per player. Then, we plot the final gain depending on the percentage of times a player has explored new cases (knowing that he had the opportunity to do it).

```
players_15 <- aggregate(file_15_R0_explore[, c("gain", "explore_yes", "explore_yes_no")],
                        list(player = file_15$player[file_15$rule == "R0"],
                             session = file_15$session[file_15$rule == "R0"]),
                        sum)
players_15$percent_explore <- players_15$explore_yes/players_15$explore_yes_no
```

We then plot the data :

```
ggplot(players_15) +
  aes(x = percent_explore, y = gain) +
  geom_point() +
  geom_smooth(method = "loess") +
  geom_smooth(method = "lm",
              col = "red")
```



We remark that :

- the player with the highest score is a player who had explored new cells every time he had the opportunity to do it,
- the player with the lowest score is a player who had not explored new cells every time he has the possibility to do it
- the linear regression line seems to indicate that the more the player explore new cases the more his final gain will be higher. However, it seems difficult to give interestings results because globally, players tend to explore new cases when they can do it. Only few of them did not adopt such a strategy.

2.3 Analysis under R1

2.3.1 Probability to get 0, 15, 99 under R1: comparaison between empirical and expected distribution

2.3.1.1 Intuitition

To optimize his profit in R1, we have the intuition that a player should play the cell which has been the most visited after the second step. At step 1 and 2, the player shoud explore new cells unless he found 99 at the 1st step.

Let's try to check if this is really the truth. For simplification, let consider a game with 5 players. The simulated grid is the following one :

simulated grid

gain = 0	gain = 0	gain = 0
gain = 0	gain = 0	gain = 15
gain = 15	gain = 15	gain = 99

At the first tour, players are supposed to explore independantly the game. The expectation of the number of coins per cell is the following one (as there are 5 players, the sum of the expectations is equal to 5) :

Information given after step 1

gain = 0 coin = 0.556	gain = 0 coin = 0.556	gain = 0 coin = 0.556
gain = 0 coin = 0.556	gain = 0 coin = 0.556	gain = 15 coin = 0.556
gain = 15 coin = 0.556	gain = 15 coin = 0.556	gain = 99 coin = 0.556

At the second tour, player who found 99 are supposed to play 99 again. The others players are supposed to explore new cells with a probability equal to $1/8$ and the exected number of coins let in each cell is thus equal to $5 \times (1 - 1/9) \times 1/8$. After the second step, we sum the expected coins at the 1st and 2nd tour, such that we obtain the expected information which is given to all players before the 3rd step :

Proba at step 2

gain = 0 coin = 0.5555	gain = 0 coin = 0.5555	gain = 0 coin = 0.5555
gain = 0 coin = 0.5555	gain = 0 coin = 0.5555	gain = 15 coin = 0.5555
gain = 15 coin = 0.5555	gain = 15 coin = 0.5555	gain = 99 coin = 1.1115

Information given after step 2

gain = 0 coin = 1.1115	gain = 0 coin = 1.1115	gain = 0 coin = 1.1115
gain = 0 coin = 1.1115	gain = 0 coin = 1.1115	gain = 15 coin = 1.1115
gain = 15 coin = 1.1115	gain = 15 coin = 1.1115	gain = 99 coin = 1.6675

We notice that the cell 99 might be the cell with the maximum expected number of coins let at the end of step2. This means that to optimize their profit in R1, players should play the cell which has been the most visited after the second step.

2.3.2 Probas to find 99 (at step 3) when playing the most visited cell (after step 2) depending on the number of coins let by players

We consider here an experiment with only 2 steps and 5 players. At each step, the player visits a new cell when he did not find 99 (such condition as in R0). At the end of the step 2, we look at the maximum number of visited cells and if it does correspond or not to the cell which contains 99.

```
simu_distrib_R1_2step <- function(game) {
  # function which return :
  # - how many times the most visited cell has been played
  # - if yes/no the most visited cell is the one which contains 99

  proba_cell <- matrix(1, 9, 5)  # 9 cells, 5 players
  nb_cell <- numeric(9)
  names(nb_cell) <- c(0, 0, 0, 0, 0, game, game, game, 99)

  # step 1
  simu_k <- rmultinom(5, size = 1, prob = rep(1/9, 9))
  nb_cell <- nb_cell + apply(simu_k, 1, sum)
  proba_cell <- proba_cell - simu_k
  # gain at 1st step
  gain <- c(0, 0, 0, 0, 0, game, game, game, 99)[apply(simu_k, 2, function(x) which(x == 1))]

  # step 2
    for (k in 1:5){
      if (proba_cell[9, k] == 0) {
        nb_cell[9] <- nb_cell[9] + 1
        } else {
          if (gain[k] == 50) {
            cell_played_50 <- (6:8)[which(proba_cell[6:8, k]) == 1]
```

```
            nb_cell[cell_played_50] <- nb_cell[cell_played_50] + 1
            } else {
              # 2nd simulation
              simu_k <- rmultinom(1, size = 1, prob = proba_cell[, k])
              nb_cell <- nb_cell + simu_k
              }
        }
    }
  # question 1
  max_value <- max(nb_cell)
  # question 2 : if several max, we choose randomly one
  # is the cell the most visited is the good one ?
  ind_value <- sample(rep(which(nb_cell == max_value), 2), 1)
  return(c(nb_max = max_value, found = (ind_value == 9)))
}
```

We simulate it several times :

```
res_sim_R1 <- replicate(100000, simu_distrib_R1_2step(15))
```

We observe that at the end of step 2, when the most visited cell has been visited 6 or more than 6 times, the player is certain to find 99 in that cell. However, when the maximum number of visited cell is equal to 2, the proba to find 99 is equal to 15.4% $(5293/(5293 + 28657))$.

```
table(res_sim_R1[1, ], res_sim_R1[2, ])
```

```
##
##          0     1
##   2  28723  5430
##   3  36309 10566
##   4   5958  8367
##   5    343  2754
##   6      0  1214
##   7      0   259
##   8      0    71
##   9      0     4
##   10     0     2
```

2.3.3 Simulation of the optimal behaviour of players under R1

To get the theoretical distibution of a player who optimizes his profit under R1, we do simulations. For doing that in the same condition that the experiment, we do simulations round per round. A party is in 5 steps with 5 players.

We suppose that the players adopt these two strategies :

- At step 1 and 2, they explore new cells until 99 has been found,
- After step 2, they choose to play the cell which has been the most visited :
  - If several cells have been visited the same number of times, the cell is choosen randomly.
  - If the most visited cell has already been visited (hence, player knows that it does not contain 99), the player choose to play the second most visited, then 3rd, etc...

```
simu_distrib_R1 <- function(game) {
  proba_cell <- matrix(1, 9, 5)
  nb_cell <- numeric(9)
  names(nb_cell) <- c(0, 0, 0, 0, 0, game, game, game, 99)
```

```r
gain <- matrix(0, 5, 5)

# step 1
simu_k <- rmultinom(5, size = 1, prob = rep(1/9, 9))
nb_cell <- nb_cell + apply(simu_k, 1, sum)

proba_cell <- proba_cell - simu_k
gain[, 1] <- c(0, 0, 0, 0, 0, game, game, game, 99)[apply(simu_k, 2, function(x) which(x == 1))]

tour <- 2
nb_choice <- 8

while (tour <= 5) {
  if (tour > 2) {
    # order the cells the most played at the previous step
    table_cell_max <- table(nb_cell)
    rank_cell <- NULL
    for(value_max in as.numeric(rev(names(table_cell_max)))) {
      which_value <- which(nb_cell == value_max)
      if (length(which_value) == 1) {
        rank_cell <- c(rank_cell, which_value)
        } else {
          rank_cell <- c(rank_cell, sample(which_value))
        }
    }
  }
  # simulate the result player by player
  for (k in 1:5){
    if (proba_cell[9, k] == 0) {
      gain[k, tour] <- 99
      nb_cell[9] <- nb_cell[9] + 1
      } else {
        if (gain[k, tour - 1] == 50) {
          gain[k, tour] <- 50
          cell_played_50 <- (6:8)[which(proba_cell[6:8, k]) == 1]
          nb_cell[cell_played_50] <- nb_cell[cell_played_50] + 1
          } else {

            if (tour > 2) {
              ind_rank <- 1
              while (proba_cell[rank_cell[ind_rank], k] != 1) {
                ind_rank <- ind_rank + 1
              }
              simu_k <- numeric(9)
              simu_k[rank_cell[ind_rank]] <- 1
            } else {
                simu_k <- rmultinom(1, size = 1, prob = proba_cell[, k])
            }

            nb_cell <- nb_cell + simu_k
            proba_cell[, k] <- proba_cell[, k] - simu_k
            gain[k, tour] <- c(0, 0, 0, 0, 0, game, game, game, 99)[which(simu_k == 1)]
            }
```

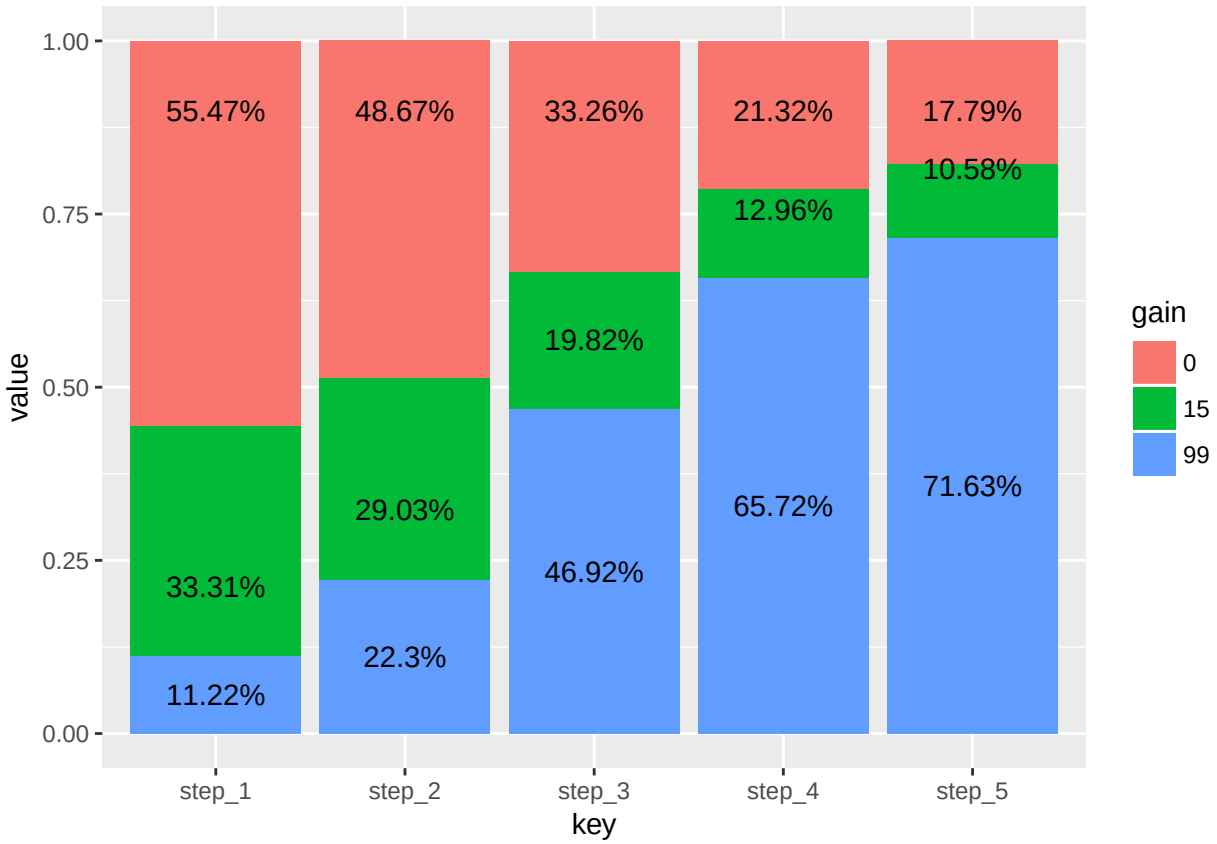
```
        }
      }
    tour <- tour + 1
    }
  return(gain)
}
```

```
distrib_th_15_R1 <- NULL
for (k in 1:20000)
  distrib_th_15_R1 <- rbind(distrib_th_15_R1, simu_distrib_R1(15))
```

We obtain this distribution. Note that at step 1 and 2, distributions are obviously identical to the ones obtained in Rule 0.

```
tab_15_R1 <- apply(distrib_th_15_R1, 2, table)/nrow(distrib_th_15_R1)
colnames(tab_15_R1) <- paste0("step_", 1:5)
gather_tab_15_R1 <- gather(as.data.frame(tab_15_R1))
gather_tab_15_R1$gain <- factor(rep(c(0, 15, 99), 5))
gather_tab_15_R1$pos <- NULL
gather_tab_15_R1$pos[gather_tab_15_R1$gain == "0"] <- 0.9
gather_tab_15_R1$pos[gather_tab_15_R1$gain == "99"] <-
  gather_tab_15_R1$value[gather_tab_15_R1$gain == "99"]/2
gather_tab_15_R1$pos[gather_tab_15_R1$gain == "15"] <-
  0.1 + gather_tab_15_R1$value[gather_tab_15_R1$gain == "99"]
```

```
ggplot(gather_tab_15_R1, aes(y = value, x = key, fill = gain)) +
  geom_bar(stat = "identity") +
  geom_text(aes(x = key, y = pos, label = paste0(round(value * 100, 2) ,"%")),
              size = 4)
```



We compare below the empirical distribution with the theoretical one obtained previously. We use a χ^2 test which consists in comparing at each step the empirical distribution of 0, 15, 99 to the theoretical one computed previously. At step 2, players behave normally. At step 3, 4 and 5, the p-values of the test are very close to 5% which indicate that players behave not completely as players who optimize their profit, but the differences are not so big.

```
file_15_R1 <- file_15[file_15$rule == "R1", ]
chisq.test(table(as.factor(file_15_R1[, "gain_2"])),
           p = tab_15_R1[, 2])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R1[, "gain_2"]))
## X-squared = 2.9428, df = 2, p-value = 0.2296
```

```
chisq.test(table(as.factor(file_15_R1[, "gain_3"])),
           p = tab_15_R1[, 3])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R1[, "gain_3"]))
## X-squared = 6.7202, df = 2, p-value = 0.03473
```

```
chisq.test(table(as.factor(file_15_R1[, "gain_4"])),
           p = tab_15_R1[, 4])
```

```
##
```

```
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R1[, "gain_4"]))
## X-squared = 4.8324, df = 2, p-value = 0.08926
```

```
chisq.test(table(as.factor(file_15_R1[,"gain_5"])),
           p = tab_15_R1[, 5])
```

```
##
##  Chi-squared test for given probabilities
##
## data:  table(as.factor(file_15_R1[, "gain_5"]))
## X-squared = 8.1895, df = 2, p-value = 0.01666
```

2.3.4 Final gain under R1: comparaison between empirical and expected distribution

Thanks to the previous simulation, we can obtain the statistical distibution of the final gain in R1 when player adopts a optimal behaviour.

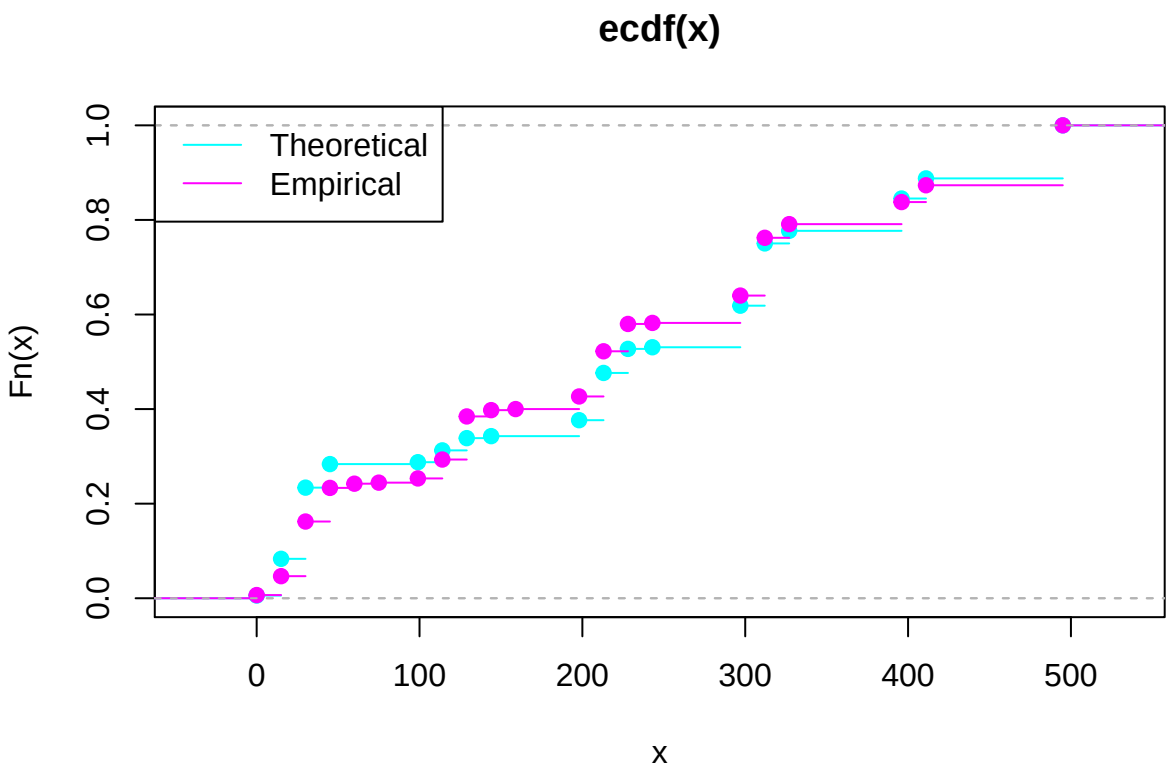
```
final_gain_15_R1 <- apply(distrib_th_15_R1, 1, sum)
```

The plot of the empirical cumulative distribution function ("theoretical" versus "empirical") does not show a big difference between the two distributions.

```
cumsum(prop.table(table(final_gain_15_R1)))
```

```
##       0      15      30      45      99     114     129     144     198
## 0.00571 0.08335 0.23390 0.28366 0.28766 0.31265 0.33853 0.34278 0.37650
##     213     228     243     297     312     327     396     411     495
## 0.47644 0.52727 0.53075 0.61867 0.75033 0.77695 0.84531 0.88781 1.00000
```

```
plot.ecdf(final_gain_15_R1, col = "cyan")
plot.ecdf(file_15_R1[, "gain"], col = "magenta", add = T)
legend("topleft", legend = c("Theoretical", "Empirical"), lty = 1, col = c("cyan", "magenta"))
```



The following test confirms the previous plot. Hereafter, the test rejects the hyptohesis of equality of the two dsitributions, but the this difference is not so big.

```
ks.test(x = final_gain_15_R1,
        y = file_15_R1[, "gain"])
```

```
## Warning in ks.test(x = final_gain_15_R1, y = file_15_R1[, "gain"]): p-value
## will be approximate in the presence of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  final_gain_15_R1 and file_15_R1[, "gain"]
## D = 0.071678, p-value = 0.02004
## alternative hypothesis: two-sided
```

2.3.5 Behaviours among the players under R1

We detect players who copy other players at each step. For doing this, we need to know which is the cell the most visited at the end of a step. We create a matrix which contains for each of the 9 cells, the number of times it has been visited at the end of a step for a given round.

For doing this, we need to count the number of visits per cell at each step, for given players (A or B) per round and per session:

```
file_15_R1$round_p <- paste0(file_15_R1$round, "_", substr(file_15_R1$player, 1, 1),
                             "_s", substr(file_15_R1$session, 9, 10))
```

Hence, we create a function which could be useful later:

```
cell_visited <- function(file) {
  # this function returns a data.frame. A row corresponds
```

```
  # to a step of a round and gives the number of times a
  # cell has been visited.
  visited_cell <- data.frame(round_p = rep(unique(file$round_p), each = 5),
                             step = rep(paste0("step_", 1:5)),
                             c_0_0 = 0, c_0_1 = 0, c_0_2 = 0,
                             c_1_0 = 0, c_1_1 = 0, c_1_2 = 0,
                             c_2_0 = 0, c_2_1 = 0, c_2_2 = 0)
  # the name of the game (a game =  one session + one kind of player A or B)
  unique_party <- unique(visited_cell$round_p)
  # we fill the matrix game after game
  for (k in 1:length(unique_party)) {
    extract_k <- file_15_R1[file_15_R1$round_p == unique_party[k], ]
    for (i in 1:5) {
      visited_cell[5 * (k - 1) + 1:5, paste0("c_", extract_k$cell_1)[i]] <-
        visited_cell[5 * (k - 1) + 1:5, paste0("c_", extract_k$cell_1)[i]] + 1
      visited_cell[5 * (k - 1) + 2:5, paste0("c_", extract_k$cell_2)[i]] <-
        visited_cell[5 * (k - 1) + 2:5, paste0("c_", extract_k$cell_2)[i]] + 1
      visited_cell[5 * (k - 1) + 3:5, paste0("c_", extract_k$cell_3)[i]] <-
        visited_cell[5 * (k - 1) + 3:5, paste0("c_", extract_k$cell_3)[i]] + 1
      visited_cell[5 * (k - 1) + 4:5, paste0("c_", extract_k$cell_4)[i]] <-
        visited_cell[5 * (k - 1) + 4:5, paste0("c_", extract_k$cell_4)[i]] + 1
      visited_cell[5 * (k - 1) + 5:5, paste0("c_", extract_k$cell_5)[i]] <-
        visited_cell[5 * (k - 1) + 5:5, paste0("c_", extract_k$cell_5)[i]] + 1
    }
  }
  return(visited_cell)
}
```

We apply the previous function to our data :

```
cell_visited_R1 <- cell_visited(file_15_R1)
```

Once we kwow how many times the cells have been visited at each step, one could say if a player decides "yes" or "not" to play the most visited cell. We do the following assumption :

- If player has already played the most visited cell and knows that it does not contain the value 99, he is supposed to play the second most visited cell, then the third, etc.

For doing this, we create a function (which could be also used later):

```
copy_or_not <- function (file, cell_visited) {
  # this function takes as input a file with the raw data and the
  # file which gives the number of times a cell has been visited at each step
  # it returns a data.frame with 3 nex columns corresponding to "yes" or "no"
  # a player has played one of the most visited cell

  file$copy3 <- NA
  file$copy4 <- NA
  file$copy5 <- NA

  for (k in 1:nrow(file)) {
    # we loop on the steps
    for (j in 3:5) {
      # at step j, we look at the cells the most visited cells
      order_step <- rev(
        sort(cell_visited[which(cell_visited$round_p == file$round_p[k])[j - 1], 3:11]))
```

```
      choice_1 <- substr(names(order_step[1]), 3, 5)
      if (!(99 %in% file[k, paste0("gain_", 1:(j - 1))])) {
        if (file[k, paste0("cell_", j)] == choice_1 &
            !(choice_1 %in% file[k, paste0("cell_", 1:(j - 1))])) {
          file[k, paste0("copy", j)] <- "yes"
          } else {
            if (choice_1 %in% file[k, paste0("cell_", 1:(j - 1))]) {
              # we look at the second choice
              choice_2 <- substr(names(order_step[2]), 3, 5)
              if (file[k, paste0("cell_", j)] == choice_2 &
                  !(choice_2 %in% file[k, paste0("cell_", 1:(j - 1))])) {
                file[k, paste0("copy", j)] <- "yes"
                } else {
                  if (choice_2 %in% file[k, paste0("cell_", 1:(j - 1))]) {
                    # we look at the 3rd choice
                    choice_3 <- substr(names(order_step[3]), 3, 5)
                    if (file[k, paste0("cell_", j)] == choice_3 &
                        !(choice_3 %in% file[k, paste0("cell_", 1:(j - 1))])) {
                      file[k, paste0("copy", j)] <- "yes"
                      } else {
                        if (choice_3 %in% file[k, paste0("cell_", 1:(j - 1))]) {
                          # we look at the 4th choice
                          choice_4 <- substr(names(order_step[4]), 3, 5)
                          if (file[k, paste0("cell_", j)] == choice_4 &
                              !(choice_4 %in% file[k, paste0("cell_", 1:(j - 1))]) ) {
                            file[k, paste0("copy", j)] <- "yes"
                            } else {
                              file[k, paste0("copy", j)] <- "no"
                              }
                          } else {
                            file[k, paste0("copy", j)] <- "no"
                          }
                        }
                    } else {
                      file[k, paste0("copy", j)] <- "no"
                      }
                }
            } else {
              file[k, paste0("copy", j)] <- "no"
            }
          }
      }
    }
  }
  return(file)
}
```

We apply our data to the previous function :

```
file_15_R1_copy <- copy_or_not(file_15_R1, cell_visited_R1)
```

For representing the data, we first tidy the data:

```
gather_file_15_R1 <- file_15_R1_copy %>%
  select(copy3, copy4, copy5, gain) %>%
```

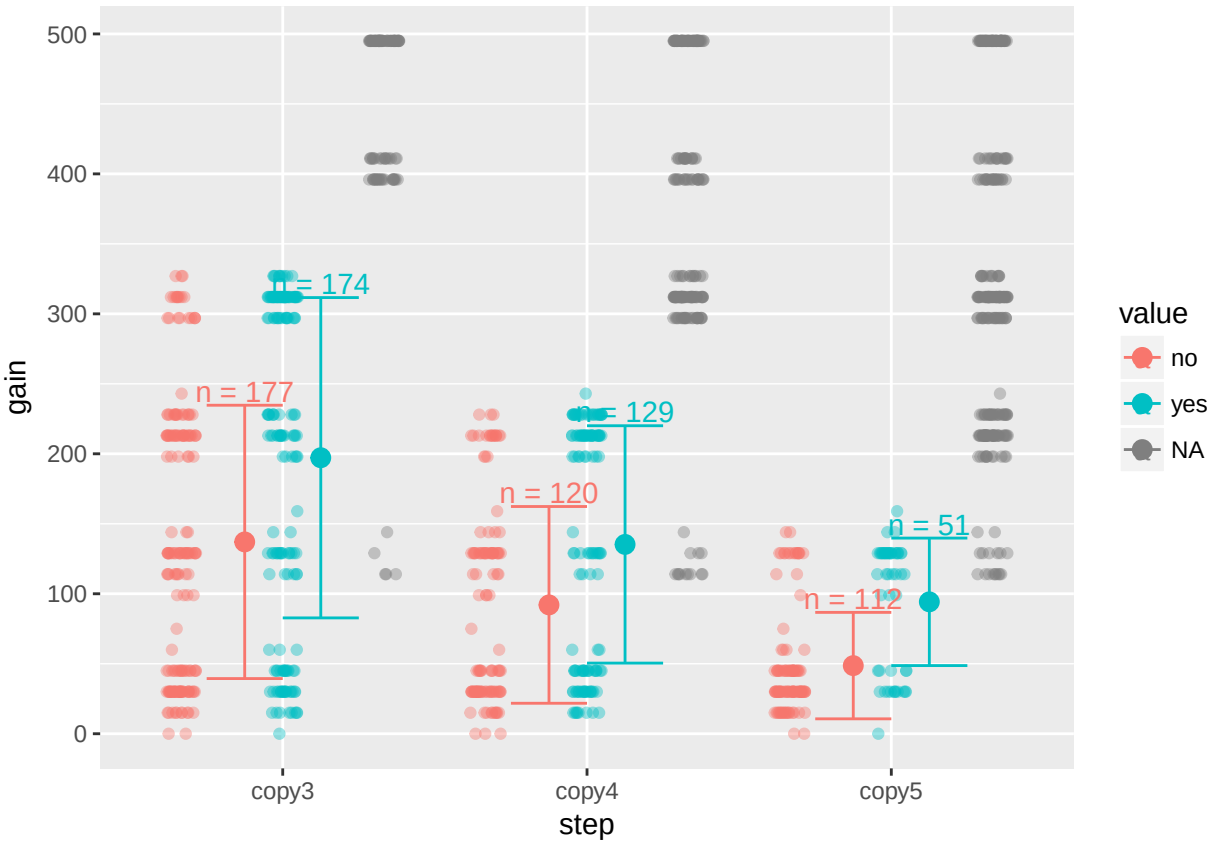
```
  gather(gain)
names(gather_file_15_R1)[2] <- "step"
```

We compute the mean and standard deviation obtained at each step and depending on the fact that a player explored new cases or not :

```
stat_gather_file_15_R1 <- merge(aggregate(gather_file_15_R1$gain,
                                          by = list(step = gather_file_15_R1$step,
                                                    value = gather_file_15_R1$value), mean),
                                merge(aggregate(gather_file_15_R1$gain,
                                                by = list(step = gather_file_15_R1$step,
                                                          value = gather_file_15_R1$value), sd),
                                      aggregate(gather_file_15_R1$gain,
                                                by = list(step = gather_file_15_R1$step,
                                                          value = gather_file_15_R1$value), length),
                                      by = c("step", "value")),
                                by = c("step", "value"))
names(stat_gather_file_15_R1)[3:5] <- c("gain", "sd", "length")
```

We plot in y-axis the gain per round and in x-axis the steps. We represent in blue (resp. in red) the gain per round obtained when a player played a visited cell cell (resp. when he did not visit) knowing that the player had the opportunity to do it. We plot the gain per round in grey obtained by people who did not have the opportunity to visit (that means that they know where the cell 99 is located):

```
ggplot(gather_file_15_R1, aes(x = step, y = gain, colour = value)) +
  geom_jitter(alpha = I(2/5),
              position = position_jitterdodge(jitter.width = 0.1, dodge.width = 1)) +
  geom_errorbar(data = stat_gather_file_15_R1, aes(ymin = gain - sd, ymax = gain + sd, colour = value),
                position = position_dodge(width = 0.5)) +
  geom_point(data = stat_gather_file_15_R1, aes(y = gain, x = step, colour = value),
             position = position_dodge(width = 0.5), size = 3) +
  geom_text(data = stat_gather_file_15_R1, aes(y = 10 + gain + sd, label = paste0("n = " , length)),
            position = position_dodge(width = 0.5))
```



To test the hypothesis of equality of means at each step, we use a non parametric test called "Kruskal-Wallis Rank Sum Test". It shows us that the differences of the means between players who visit and players who do not visit are significant.

```
(kruskal.test(gather_file_15_R1$gain[gather_file_15_R1$step == "copy3"] ~
                           factor(gather_file_15_R1$value[gather_file_15_R1$step == "copy3"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15_R1$gain[gather_file_15_R1$step == "copy3"] by factor(gather_file_15_R1$value[ga
## Kruskal-Wallis chi-squared = 26.558, df = 1, p-value = 2.558e-07
```

```
(kruskal.test(gather_file_15_R1$gain[gather_file_15_R1$step == "copy4"] ~
                           factor(gather_file_15_R1$value[gather_file_15_R1$step == "copy4"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15_R1$gain[gather_file_15_R1$step == "copy4"] by factor(gather_file_15_R1$value[ga
## Kruskal-Wallis chi-squared = 19.008, df = 1, p-value = 1.302e-05
```

```
(kruskal.test(gather_file_15_R1$gain[gather_file_15_R1$step == "copy5"] ~
                           factor(gather_file_15_R1$value[gather_file_15_R1$step == "copy5"])))
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  gather_file_15_R1$gain[gather_file_15_R1$step == "copy5"] by factor(gather_file_15_R1$value[ga
```

```
## Kruskal-Wallis chi-squared = 26.678, df = 1, p-value = 2.403e-07
```

2.3.5.1 The effect of visiting (or not) on the final gain

We count the number of times a player has visited the cell the most visited when he had the opportunity to do it:

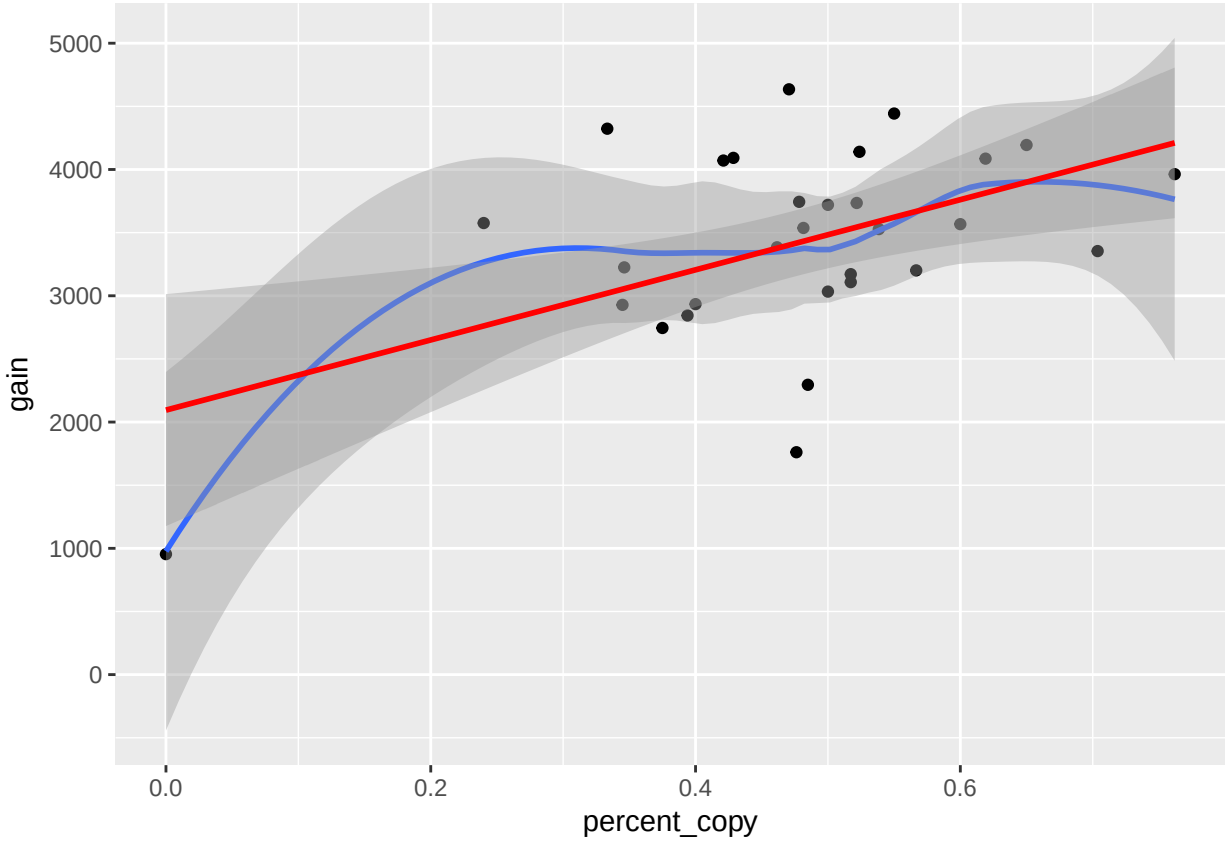
```
file_15_R1_copy$copy_yes <- apply(file_15_R1_copy[, c("copy3", "copy4", "copy5")],
                                  1, function(x) length(which(x == "yes")))
file_15_R1_copy$copy_no <- apply(file_15_R1_copy[, c("copy3", "copy4", "copy5")],
                                  1, function(x) length(which(x == "no")))
file_15_R1_copy$copy_yes_no <- file_15_R1_copy$copy_yes + file_15_R1_copy$copy_no
```

We aggregate the data to obtain statistics per player in a session :

```
players_15_R1 <- aggregate(file_15_R1_copy[, c("gain", "copy_yes", "copy_yes_no")],
                           list(player = file_15_R1_copy$player[file_15_R1_copy$rule == "R1"],
                                session = file_15_R1_copy$session[file_15_R1_copy$rule == "R1"]),
                           sum)
players_15_R1$percent_copy<- players_15_R1$copy_yes/players_15_R1$copy_yes_no
```

We plot the gains depending on the strategy:

```
ggplot(players_15_R1) +
  aes(x = percent_copy, y = gain) +
  geom_point() +
  geom_smooth(method = "loess") +
  geom_smooth(method = "lm",
              col = "red")
```



We constat that the more the player adopts a strategy of copying, the more he obtains an important final gain.

2.4 Analysis under R2

TBD

3 Statistical analysis of Stigmer 50

TBD