

# WCST analysis

## Check data

The size of the raw data table is (number of moves x number of variables):

```
## [1] 269 11
```

There should be 4 types of rules: c, n, s and spxxxx (x stands for A, B, C or D). If there is more than 4, calculations based on the rules will be wrong!

```
##      c      n      s spABCD
##     40     52     60     117
```

There should be  $24+1=25$  types of source cards if we use unambiguous cards or  $64-4+1=61$  types if we use ambiguous cards. The number of source card types:

```
## [1] 61
```

All source cards should have a 4-digit code. The source card codes and frequencies were:

```
## cg1 cy1 cy4 qb1 qg1 qy4 sb3 sb4 sg3 sg4 sr1 sy1 tg2 tg3 tr2 ty1 ty3 cb3
##   5   2   3   5   3   4   5   3   3   4   3   3   4   6   4   3   6   6
## cg4 qb2 qg4 qr2 wh5 cb1 cg3 cr1 cr2 cr3 cy2 cy3 qb3 qb4 qg2 qg3 qr1 qr3
##   3   3   3   5  79   3   3   2   4   3   7   2   4   2   2   2   2   3
## qr4 qy1 qy2 sb1 sb2 sg1 sr3 sr4 sy2 sy3 tb1 tb2 tb4 tg1 tg4 tr3 tr4 ty4
##   2   3   2   3   3   4   2   4   2   5   4   2   3   4   2   3   3   3
## cb2 cg2 cr4 sr2 sy4 tb3 ty2
##   2   1   2   2   1   1   2
```

There should be 4 types of target cards, with the codes: tr1A, sg2B, qy3C, cb4D. Target card codes and frequencies were:

```
##      cb4D      qy3C      sg2B      tr1A div1undefined
##      51      97      72      48      1
```

## Variables

### Additional variables per move

From the raw data, I calculated some additional variables, such as:

- Think time: total time between two drop action minus the time spent with drag-and-drop
- Current rule: this is the same as “category”, but recoded with numbers, where s, c, n and sp have the codes: 1, 2, 3 and 4, respectively.
- Applied rule: the rule that the move conforms to. THIS WON'T BE CORRECT UNTIL THE SOURCE CARD CODE DOES NOT INCLUDE THE INDEX FOR THE SPATIAL RULE
- Number of applied rules: integer from 0 to 4 (actually 3, because we excluded the key cards from the deck)
- Random search: this is TRUE if none of the rules applies to the current rule (if the number of applied rules = 1)
- Perseveration error: after a “success”, we check if the participant keeps using the same rule (if the applied rule - or one of the applied rules - is the same as the current rule in the successful step) for any number of moves

RANDOM SEARCH: NOT STICKING TO ONE RULE FOR MORE THAN 2 MOVES; PAYING ATTENTION TO THE FEEDBACK

IGNORING FEEDBACK

### An example: Data for participant 242040

```
## [1] tar_card    src_card    match        current_rule applied_rule
## [6] randsearch   persev
## <0 rows> (or 0-length row.names)
```

### Variables per participant

From the per-move variables, I calculated averaged and summed values for each participant, separately for each rule:

- success
- failure
- number of moves
- number of correct moves
- number of incorrect moves
- average time between two drop actions
- average move time
- average think time
- time spent with the task (sec)

```
## Source: local data frame [29 x 6]
## Groups: condition, part [8]
##
##   condition  part category success_bin failure_bin moves_nb
##   (fctr)    (int)  (fctr)      (int)      (int)      (dbl)
## 1    wcstf 178897      c           0           0           3
## 2    wcstf 178897      n           0           1           3
## 3    wcstf 178897      s           0           1           3
## 4    wcstf 178897  spABCD           0           1           9
## 5    wcstfc 259228      c           0           1           3
## 6    wcstfc 259228      n           0           0           3
## 7    wcstfc 259228      s           0           1           3
## 8    wcstfc 259228  spABCD           0           1           9
## 9    wcstfc 762799      c           1           0           8
## 10   wcstfc 762799      n           1           0           8
## ..      ...      ...      ...      ...      ...
##
## Source: local data frame [29 x 6]
##
##   correctmoves_nb incorrectmoves_nb totaltime_avg movetime_avg
##   (int)          (int)          (dbl)          (dbl)
## 1           1           1       1748.000       792.0000
## 2           1           2       1562.333       650.0000
## 3           1           2       1926.000       885.0000
## 4           2           7       2034.000       858.4444
## 5           0           3       2181.000       521.6667
## 6           1           1       1757.000       802.5000
## 7           2           1       1612.000       611.0000
```

```
## 8          1          8      1825.667      646.0000
## 9          8          0      4605.750      2035.1250
## 10         7          1      3155.875      1613.1250
## ..      ...      ...      ...      ...
## Variables not shown: thinktime_avg (dbl), time_spent (dbl)
```

- *Would perseveration be interesting? Do we need it?*
- *Distribution of rules used?*

## Variables per condition

I coded the following variables, separately for each rule:

- Number of participants
- Number of solvers
- Number of nonsolvers
- Percentage of solvers
- Average number of moves
- Average time spent with the task

*Other ideas?*

```
## Source: local data frame [16 x 6]
## Groups: condition [4]
##
##   condition category partic_nb solvers_nb nonsolvers_nb solvers_perc
##   (fctr)    (fctr)    (int)    (int)    (int)    (dbl)
## 1    wcstf      c        0        0        0        NA
## 2    wcstf      n        1        0        1    0.00000
## 3    wcstf      s        1        0        1    0.00000
## 4    wcstf    spABCD      1        0        1    0.00000
## 5    wcstfc     c        2        1        1    50.00000
## 6    wcstfc     n        1        1        0   100.00000
## 7    wcstfc     s        1        0        1    0.00000
## 8    wcstfc    spABCD      2        1        1    50.00000
## 9    wcstc     c        3        1        2    33.33333
## 10   wcstc     n        3        1        2    33.33333
## 11   wcstc     s        3        1        2    33.33333
## 12   wcstc    spABCD      0        0        0        NaN
## 13   wcst      c        1        1        0   100.00000
## 14   wcst      n        1        1        0   100.00000
## 15   wcst      s        1        1        0   100.00000
## 16   wcst    spABCD      0        0        0        NaN
## Source: local data frame [16 x 2]
##
##   nbofmoves_avg timespent_avg
##   (dbl)        (dbl)
## 1    3.000000    3.49600
## 2    3.000000    4.68700
## 3    3.000000    5.77800
## 4    9.000000   18.30600
## 5    5.500000   21.69450
## 6    5.500000   14.38050
## 7   10.000000   23.60800
## 8   23.000000  153.57000
```

## 9	5.333333	11.56267
## 10	8.666667	20.02733
## 11	8.000000	16.83700
## 12	12.000000	19.98400
## 13	11.000000	22.56600
## 14	7.000000	15.70850
## 15	14.000000	31.35700
## 16	30.000000	155.06700

## Hypotheses

In our plans we had the following testable hypotheses:

1. Going through the standard rules first makes the task of finding the index rule harder
  - a) Condition 1 (WCST) will be more difficult than condition 2 (WCSTF)
  - b) Condition 3 (WCSTC) will be more difficult than condition 4 (WCSTFC)
  - c) Condition 1 (WCST) will be more difficult than condition 4 (WCSTFC)
2. Using the standard cards for the index rule makes the task of finding the index rule harder
  - a) Condition 1 (WCST) will be more difficult than condition 3 (WCSTC)
  - b) Condition 2 (WCSTF) will be more difficult than condition 4 (WCSTFC)
  - c) Condition 1 (WCST) will be more difficult than condition 4 (WCSTFC)
3. Some participants will report having an Heureka! moment, so this task can be categorized as an insight task
4. Is impasse associated with inactivity in this task? Is the average move time longer for the spatial task than for the other tasks?

*Would you like to add something else? Can we justify Hypothesis 1/c and 2/c?*

## Analyses

### Exclusion of participants

*What exclusion criteria do we want to use?*

- Exclude participants who did not finish the experiment?
- Exclude participants who did not find out the standard rules?
- Exclude participants who stay inactive for more than a certain amount of time?
- Exclude participants who respond too quickly? Who did not do the questionnaire Who were inactive for more than 1 minute For spatial rule: who did not solve the other rules if std and sp rule is last: if they only make one mistake if std and sp rule is first: if they only make three mistake if moon: if they make 0 mistake

### Statistical tests

For deciding whether solving the task in one condition is more difficult than solving it in another, we can use two variables: solution rates and solution times. As for the solution rates, we can construct a four-by-two contingency table (number of solvers and non-solvers in the four conditions). Analyzing a larger than two-by-two contingency table is possible with a Chi-square test, but it is only useful if the categories defining the rows are arranged in a natural order with equal spacing between rows. This is not true in our case, so we will have to analyze two-by-two contingency tables instead, separately for Hypothesis 1 and Hypothesis 2.

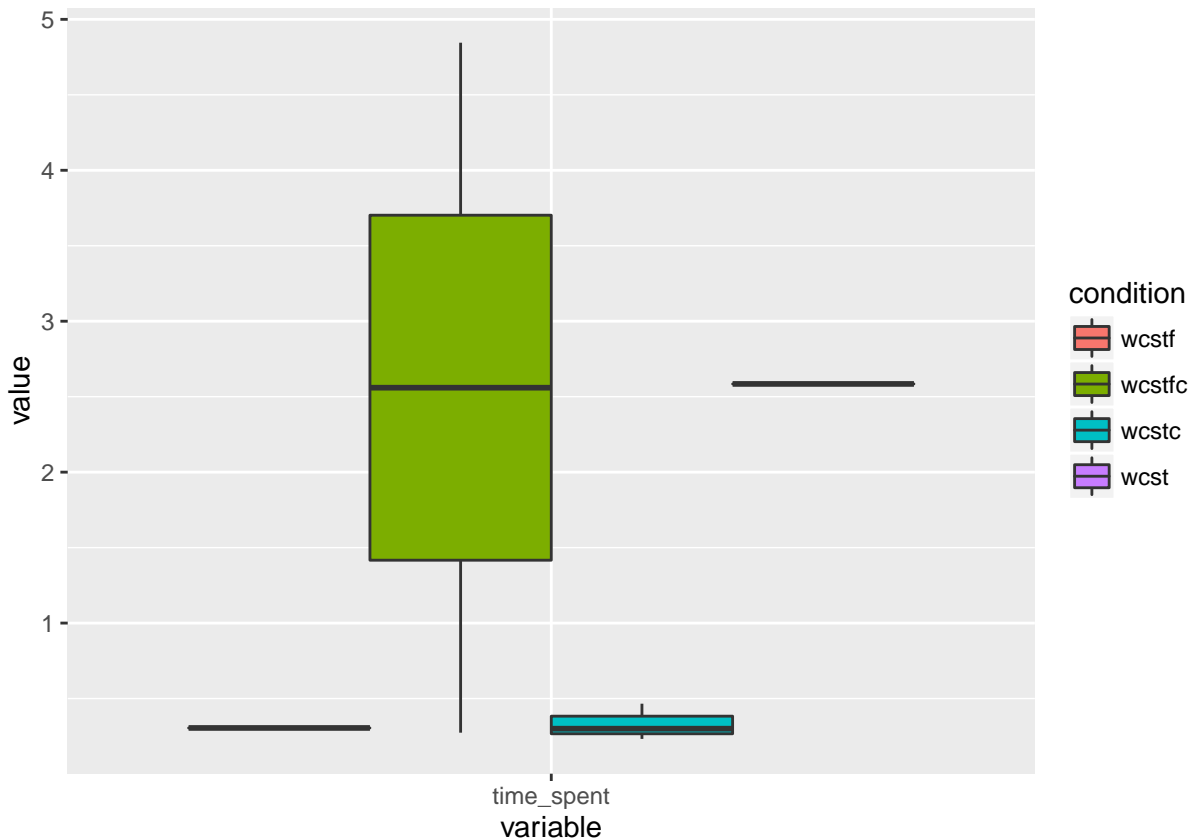
We chose Fisher's exact test, which is supposed to be more exact than the Chi-square test, especially if the cells in the contingency table contain small values. OR BONFERRONI CORRECTION

BINARY LOGISTIC REGRESSION, persev could be a predictor less persev -> more likely to find sp rule

Solution rates in the four conditions:

```
## Source: local data frame [4 x 3]
## Groups: condition [4]
##
##   condition solvers_nb nonsolvers_nb
##   (fctr)      (int)      (int)
## 1   wcstf         0         1
## 2   wcstfc        1         1
## 3   wcstc         0         0
## 4   wcst          0         0
```

Solution time is calculated for each participant as the time spent with trying to solve the sequence rule (in minutes). We analyzed the solution time with a two-way ANOVA. Median solution times for the four conditions are shown by the box plots in the following figure:



## Number of solvers

### Does the order of rules influence the difficulty of finding the sequence rule?

We analyzed the contingency table containing the number of solvers and non-solvers in condition WCST and WCSTF and then separately, for condition WCSTC and WCSTFC. A  $p < 0.05$  means that the row/column association is statistically significant.

```

##
## Fisher's Exact Test for Count Data
##
## data:  SR1234[c(1, 3), c(2, 3)]
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##      0 Inf
## sample estimates:
## odds ratio
##          0

##
## Fisher's Exact Test for Count Data
##
## data:  SR1234[c(2, 4), c(2, 3)]
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##      0 Inf
## sample estimates:
## odds ratio
##          0

```

### Does the order of rules influence the difficulty of finding the sequence rule?

We analyzed the contingency table containing the number of solvers and non-solvers in condition WCST and WCSTC and then separately, for condition WCSTF and WCSTFC. A  $p < 0.05$  means that the row/column association is statistically significant.

```

##
## Fisher's Exact Test for Count Data
##
## data:  SR1234[c(1, 2), c(2, 3)]
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.00000 77.90627
## sample estimates:
## odds ratio
##          0

##
## Fisher's Exact Test for Count Data
##
## data:  SR1234[c(3, 4), c(2, 3)]
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##      0 Inf
## sample estimates:
## odds ratio
##          0

```

## Solution time

We checked whether the data was normally distributed with Kolmogorov-Smirnov test:

```
##
## One-sample Kolmogorov-Smirnov test
##
## data: ST1
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided

##
## One-sample Kolmogorov-Smirnov test
##
## data: ST2
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided

##
## One-sample Kolmogorov-Smirnov test
##
## data: ST3
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided

##
## One-sample Kolmogorov-Smirnov test
##
## data: ST4
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

If the data is normally distributed, we use ANOVA, if it is not, we use Wilcoxon.

```
## Warning in wilcox.test.default(ST1, ST2, alternative = "two.sided", paired
## = FALSE, : Requested conf.level not achievable

##
## Wilcoxon rank sum test
##
## data: ST1 and ST2
## W = 1, p-value = 1
## alternative hypothesis: true location shift is not equal to 0
## 0 percent confidence interval:
## 488983 488983
## sample estimates:
## difference in location
## 488983
```