

# Analysis

Niclas Dobbertin

October 28, 2023

## Contents

<b>1 Imports</b>	<b>1</b>
<b>2 Constants</b>	<b>1</b>
<b>3 Import Data</b>	<b>2</b>
3.1 Conditions . . . . .	2
3.2 Data . . . . .	2
3.3 Useful Subdata . . . . .	2
<b>4 Basic statistics</b>	<b>2</b>
4.1 Total percent correct . . . . .	2

## 1 Imports

```
import pandas as pd
from pathlib import Path
from pprint import pprint
```

```
import tools
```

## 2 Constants

```
data_path = Path("/home/niclas/repos/uni/master_thesis/experiment/data")
```

```
procedures = ["1", "2", "3", "4", "5", "6", "overall"]
```

## 3 Import Data

### 3.1 Conditions

```
conditions = [x.stem for x in data_path.iterdir() if x.is_dir()]
conditions
```

```
random fixed blocked
```

### 3.2 Data

```
data = {}
for condition in conditions:
    data[condition] = {}
    for vp in (data_path / condition).iterdir():
        data[condition][vp.stem] = tools.unpickle(vp / "vp.pkl")
```

```
None
```

### 3.3 Useful Subdata

```
# data_correct = {conditions[0]: {}, conditions[1]: {}, conditions[2]: {}}
pass
# for condition in conditions:
#     data_correct[condition] = None
```

```
None
```

## 4 Basic statistics

### 4.1 Total percent correct

To find out how well VP solved the task, we calculate the accuracy for train and test phase.

```
condition = "random"
df = pd.DataFrame([tools.total_accuracy(data[condition][vp], procedures) for vp in data[condition].keys()])
df
```

	train	test
vp12	0.822222	0.820000
vp19	0.966667	0.800000

```

vp15 0.973333 0.980000
vp17 0.911111 0.960000
vp20 0.906667 0.980000
vp10 0.924444 0.943333
vp16 0.957778 0.926667
vp13 0.857778 0.946667
vp18 0.962222 0.970000
vp14 0.982222 0.986667

```

Most subjects have an accuracy of over 95% in both training and test phase. Some however are notably lower, under 90% in either training or test phase, or both. This could be a systematic misunderstanding of specific equations, that are present in both, or only one of the two phases. To investigate, we look at the per procedure accuracy per subject.

```

condition = "random"
proc_accs = [
    tools.count_correct(data[condition][vp], data[condition][vp].keys(), procedures)
    for vp in data[condition].keys()
]
for vp in proc_accs:
    for proc in vp.keys():
        vp[proc] /= len(next(iter(data[condition].values())).keys())
df = pd.DataFrame(proc_accs, index=data[condition].keys())
df

```

	1	2	3	4	5	6	overall
vp12	0.992	0.592	0.392	0.976	0.960	1.000	0.016
vp19	1.000	0.992	0.000	0.576	0.992	0.992	0.848
vp15	0.992	0.992	0.960	0.392	0.592	1.000	0.928
vp17	0.392	0.968	0.584	1.000	1.000	0.992	0.648
vp20	0.992	0.376	0.952	0.976	0.976	0.560	0.784
vp10	0.968	0.360	0.592	0.984	0.984	0.992	0.712
vp16	0.976	0.600	0.376	0.976	0.992	1.000	0.752
vp13	0.384	0.960	0.928	0.560	0.992	0.968	0.568
vp18	0.976	0.976	0.960	0.392	0.600	0.984	0.904
vp14	0.992	0.976	0.992	0.976	0.400	0.600	0.968

We can see that most vp have around 2 procedures with accuracy of around 50%