Processing Raw Performance Data

Daniel Cabral

2022-07-14

The following code was written in Python. To run the code in R, it is necessary to install the package "reticulate" and run the function install_cond(path = "Whatever path you want").

Note that we used a custom LabVIEW program (Neumann and Thomas, 2008) to measure the distance of the ball from the target along the x and y axes. The code below imports these distances from each participant for every condition and concatenates them into one data frame. Next, the code calculates radial error and bivariate variable error for each trial. The resultant data frame can then be imported into an R Markdown file for statistical analyses.

The custom LabView program and the x- and y-axis data it yielded as well as the R Markdown file for statistical analyses can be found in supplementary materials.

Uploading packages

```
import numpy as np
import pandas as pd
import glob
```

Here, we are using data from pretest. You can easily change the path to where the data from posttests and acquisition are. You may also need to change some variables' names accordingly.

```
path = r'C:\Users\Daniel Aranha Cabral\Box\PEP lab\Teach analogy
study\Data\Data\Performance\All sample\Pretest' # change the path here

all_files = glob.glob(path + '/*.dat')

list = []

for filename in all_files:
    total_participants = pd.read_table(filename, index_col = None, header
= 0, sep = "/")
    list.append(total_participants)

total_sample = pd.concat(list, axis = 0, ignore_index = False)

total_sample.drop([0, 1, 2, 3], inplace = True)


total_sample = total_sample['Participant name'].str.split('\t', 6,
```

```
expand=True)
total_sample = total_sample.apply(lambda x: x.str.replace(',','.'))

total_sample['Y_axis'] = total_sample[1]
total_sample['X_axis'] = total_sample[4]

total_sample.drop([0,1,2,3,4,5,6], axis=1, inplace = True)

total_sample['Y_axis'] = total_sample['Y_axis'].astype(float)
total_sample['X_axis'] = total_sample['X_axis'].astype(float)

total_sample['Radial Error'] = np.sqrt((total_sample['X_axis']**2 +
total_sample['Y_axis']**2))

total_sample.reset_index(inplace = True)

total_sample.drop(['index'], axis = 1, inplace = True)

names = []
for l in all_files:
       names.append(l[103:-4])
total_sample['Participants'] = np.repeat(names, 10)
total_sample[['ID','Condition', 'Group']] =
total_sample['Participants'].str.split("_",expand=True)

total_sample.drop(['Participants'], axis = 1, inplace = True)
df = total_sample.groupby('ID').count()
soma = int(np.sum(df['Condition'])/10)
total_sample['Trial'] = np.array([i+1 for i in range(10)]*soma)

expect = []
for i in total_sample['Group']:
    if i == 'TCE' or i == 'TCA':
        expect.append('Teach')
    else:
        expect.append('Test')
total_sample['Expectation'] = expect

instruction = []
for i in total_sample['Group']:
    if i == 'TCA' or i == 'TTA':
        instruction.append('Analogy')
    else:
        instruction.append('Explicit')

total_sample['Instructions'] = instruction

total_sample = total_sample[['ID', 'Condition','Group', 'Expectation',
'Instructions', 'Trial', 'Y_axis', 'X_axis', 'Radial Error']]
```

```
total_sample['Constant_X'] = total_sample['X_axis'] - 0
total_sample['Constant_Y'] = total_sample['Y_axis'] - 0

total_sample['ID'] = total_sample['ID'].astype(int)
total_sample.sort_values("ID");

ce_x_av =  total_sample.groupby(['Condition',
'ID'])['Constant_X'].mean().reset_index()
ce_x_av = np.repeat(ce_x_av['Constant_X'], 10).reset_index()
total_sample['CE_X_AV'] = ce_x_av['Constant_X']

ce_y_av =  total_sample.groupby(['Condition',
'ID'])['Constant_Y'].mean().reset_index()
ce_y_av = np.repeat(ce_y_av['Constant_Y'], 10).reset_index()
total_sample['CE_Y_AV'] = ce_y_av['Constant_Y']

x_av = total_sample.groupby(['Condition',
'ID'])['X_axis'].mean().reset_index()
x_av = np.repeat(x_av['X_axis'], 10).reset_index()
total_sample['X_axis_av'] = x_av['X_axis']

y_av = total_sample.groupby(['Condition',
'ID'])['Y_axis'].mean().reset_index()
y_av = np.repeat(y_av['Y_axis'], 10).reset_index()
total_sample['Y_axis_av'] = y_av['Y_axis']

total_sample['BVE'] = (((total_sample['X_axis'] -
total_sample['CE_X_AV'])**2) + (total_sample['Y_axis'] -
total_sample['CE_Y_AV'])**2)
bve_av = total_sample.groupby(['Condition',
'ID'])['BVE'].mean().reset_index()
bve_av = np.repeat(bve_av['BVE'], 10).reset_index()
bve_av = np.sqrt(bve_av)
total_sample['BVE_av'] = bve_av['BVE']

total_sample.head(30)
```

```
##     ID Condition Group  ... Y_axis_av           BVE     BVE_av
## 0    1        pt   TCE  ...    1.9603    177.103652  74.908668
## 1    1        pt   TCE  ...    1.9603   7060.625558  74.908668
## 2    1        pt   TCE  ...    1.9603   2120.500008  74.908668
## 3    1        pt   TCE  ...    1.9603  10251.681688  74.908668
## 4    1        pt   TCE  ...    1.9603   1848.000795  74.908668
## 5    1        pt   TCE  ...    1.9603   1357.788333  74.908668
## 6    1        pt   TCE  ...    1.9603   3924.148507  74.908668
## 7    1        pt   TCE  ...    1.9603  17096.486965  74.908668
## 8    1        pt   TCE  ...    1.9603   6415.760321  74.908668
## 9    1        pt   TCE  ...    1.9603   5860.989564  74.908668
```

```
## 10   2        pt   TTA  ...  -36.2942  27565.186357  71.529694
## 11   2        pt   TTA  ...  -36.2942    290.222169  71.529694
## 12   2        pt   TTA  ...  -36.2942   9361.519300  71.529694
## 13   2        pt   TTA  ...  -36.2942   6812.621381  71.529694
## 14   2        pt   TTA  ...  -36.2942   3591.397275  71.529694
## 15   2        pt   TTA  ...  -36.2942    207.611446  71.529694
## 16   2        pt   TTA  ...  -36.2942    823.396149  71.529694
## 17   2        pt   TTA  ...  -36.2942    467.043375  71.529694
## 18   2        pt   TTA  ...  -36.2942    375.194893  71.529694
## 19   2        pt   TTA  ...  -36.2942   1670.778763  71.529694
## 20   5        pt   TTE  ...   16.0688   2263.607148  56.495693
## 21   5        pt   TTE  ...   16.0688  16542.200609  56.495693
## 22   5        pt   TTE  ...   16.0688   5426.814359  56.495693
## 23   5        pt   TTE  ...   16.0688    136.549737  56.495693
## 24   5        pt   TTE  ...   16.0688    149.001787  56.495693
## 25   5        pt   TTE  ...   16.0688    376.757612  56.495693
## 26   5        pt   TTE  ...   16.0688     13.868668  56.495693
## 27   5        pt   TTE  ...   16.0688    367.741643  56.495693
## 28   5        pt   TTE  ...   16.0688   2471.729902  56.495693
## 29   5        pt   TTE  ...   16.0688   4169.361630  56.495693
##
## [30 rows x 17 columns]
```