Implicit assessment in psychology:
How to work with the IAT

Ottavia M. Epifania
University of Padova

ottavia.epifania@unipd.it

June 9^{th}-10^{th}

Table of contents

What we're using today

- download R and follow the instructions
- download RStudio and follow the instructions
- 

for an easy analsyis of the data (no need to install)
- `implicitMeasures` `R` package

Disclaimer

This tutorial assumes that Inquisit is used for collecting the IAT data.

It is indeed based on data collected with Inquisit, available here

It is a Race IAT data set with observations from 142 participants.

Explicit measures (i.e., political orientation and attitudes towards Black/White people) have been collected as well

Both the shiny app and the package can be used with data collected with other administration software

DScoreApp

If you want it simple, DScoreApp is made for you!

Adavantages

- It's super easy to use
- Well documented and the maintainer is always available to help you
- You can visually inspect the results as you compute them
- There's a toy data set with which you can familiarize with the app

Disadvantages

- You have to use excel to prepare the data
- You can compute only one D score at the time
- You have to manually merge in excel the data set with explicit measures to run further analysis

Let's start

Let's take a look at DScoreApp

Let's start

Let's take a look at DScoreApp

Data set have to be arranged in 4 columns, named:

- **`participant`**: participant's IDs
- **`block`**: Mapping A and Mapping B block labels
- **`latency`**: response times
- **`correct`**: accuracy responses

The data set

From Inquisit, we usually obtain a `.dat` file:

shinyData.dat - Notepad

File Edit Format View Help

date	time	build	subject	blocknum	blockcode	trialnum	trialcode	response	correct	latency	stimulusitem1
121318	09:55	3.0.6.0	1	1	consenso	1	consenso	Disponibile a partecipare alla ricerca nei termini sopra indicati	1	27280	0
121318	09:55	3.0.6.0	1	19	WhiteBlack	1	reminder	57	1	4016	Controlla le categorie - premi SPAZIO per cominciare
121318	09:55	3.0.6.0	1	19	WhiteBlack	2	PlainWhiteleft	18	1	924	wf6.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	4	PlainBlackright	23	1	637	bm56.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	6	PlainWhiteleft	18	1	732	wm1.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	8	PlainBlackright	23	1	550	bf14.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	10	PlainBlackright	23	1	588	bm14.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	12	PlainBlackright	23	1	736	bm23.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	14	PlainWhiteleft	18	1	612	wf3.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	16	PlainWhiteleft	23	0	837	wm6.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	18	PlainWhiteleft	18	1	1380	wf2.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	20	PlainWhiteleft	18	1	1255	wm4.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	22	PlainBlackright	23	1	704	bf56.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	24	PlainBlackright	23	1	636	bf23.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	26	PlainBlackright	23	1	762	bf14.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	28	PlainBlackright	23	1	641	bm14.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	30	PlainBlackright	23	1	1096	bf56.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	32	PlainBlackright	23	1	1033	bm56.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	34	PlainWhiteleft	18	1	967	wm4.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	36	PlainWhiteleft	18	1	824	wf2.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	38	PlainWhiteleft	18	1	1139	wm1.jpg
121318	09:55	3.0.6.0	1	19	WhiteBlack	40	PlainWhiteleft	18	1	719	wf3.jpg
121318	09:55	3.0.6.0	1	21	badgood 1	reminder	57	1	3234	Controlla le categorie - premi SPAZIO per cominciare	
121318	09:55	3.0.6.0	1	21	badgood 2	Plainbadleft	18	1	1090	cattivo	
121318	09:55	3.0.6.0	1	21	badgood 4	Plainbadleft	18	1	675	fallimento	

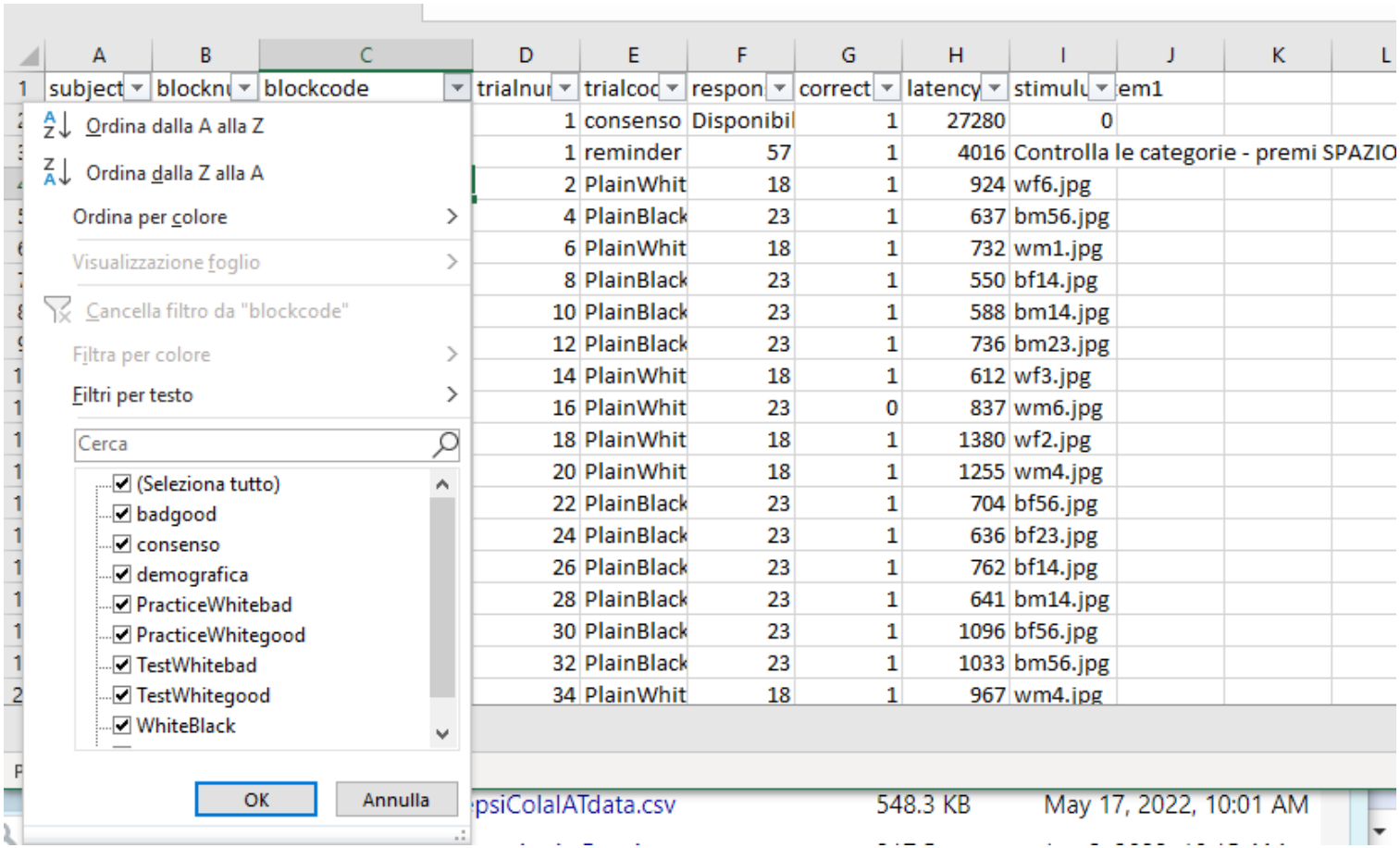
Just Copy & Paste in an Excel file! If you don't have "," as the default colum separator: Dati $\rightarrow$ Testo in colonne $\rightarrow$ Delimitato $\rightarrow$ virgola

Just delete the unnecessary columns:

- `date`
- `time`
- `build`

The first column of interest is `blockcode`. Use the filter function to see all the possible values of the column

blockcode



subject	blockn	blockcode	trialnur	trialcod	respon	correct	latency	stimulu	em1
			1	consenso	Disponibil	1	27280	0	
			1	reminder	57	1	4016	Controlla le categorie - premi SPAZIO	
			2	PlainWhit	18	1	924	wf6.jpg	
			4	PlainBlack	23	1	637	bm56.jpg	
			6	PlainWhit	18	1	732	wm1.jpg	
			8	PlainBlack	23	1	550	bf14.jpg	
			10	PlainBlack	23	1	588	bm14.jpg	
			12	PlainBlack	23	1	736	bm23.jpg	
			14	PlainWhit	18	1	612	wf3.jpg	
			16	PlainWhit	23	0	837	wm6.jpg	
			18	PlainWhit	18	1	1380	wf2.jpg	
			20	PlainWhit	18	1	1255	wm4.jpg	
			22	PlainBlack	23	1	704	bf56.jpg	
			24	PlainBlack	23	1	636	bf23.jpg	
			26	PlainBlack	23	1	762	bf14.jpg	
			28	PlainBlack	23	1	641	bm14.jpg	
			30	PlainBlack	23	1	1096	bf56.jpg	
			32	PlainBlack	23	1	1033	bm56.jpg	
			34	PlainWhit	18	1	967	wm4.jpg	

psiColalATdata.csv 548.3 KB May 17, 2022, 10:01 AM

`blockcode`

The blocks we need are the "critical" blocks of the IAT:

- `PracticeWhitegood`
- `TestWhitegood`
- `PracticeWhitebad`
- `TestWhitebad`

blockcode

The blocks we need are the "critical" blocks of the IAT:

- PracticeWhitegood
- TestWhitegood
- PracticeWhitebad
- TestWhitebad

White-Good/Black-Bad Condition (MappingA)

blockcode

The blocks we need are the "critical" blocks of the IAT:

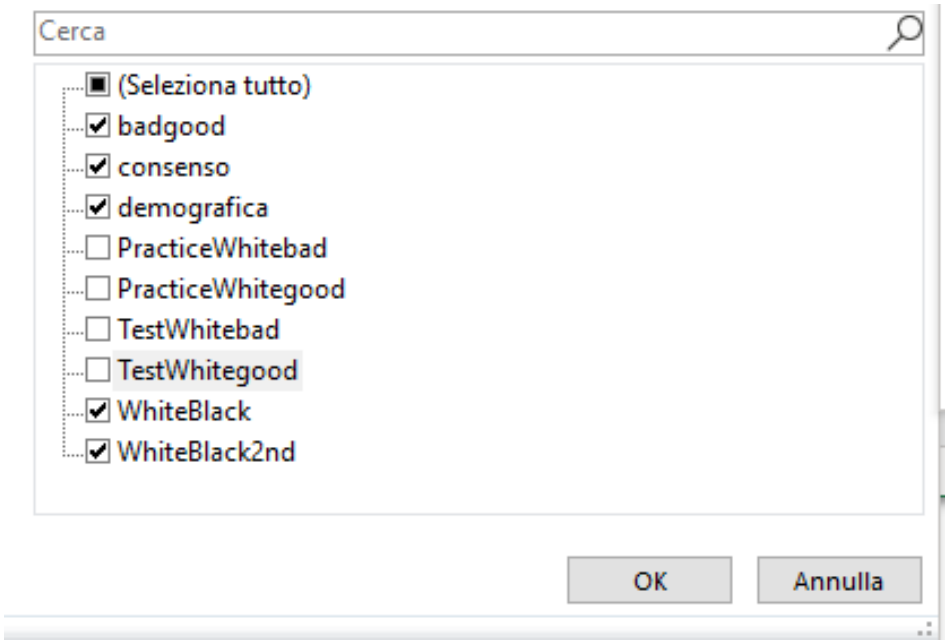
- `PracticeWhitegood`
- `TestWhitegood`
- `PracticeWhitebad`
- `TestWhitebad`

White-Good/Black-Bad Condition (MappingA)

Black-Good/White-Bad Condition (MappingB)

Just select those.... by eliminating all other blocks!

From the filter on the column `blockcode`, select all blocks but `PracticeWhitegood`, `TestWhitegood`, `PracticeWhitebad`, `TestWhitebad`:



`blockcode`

Highlight and delete all the rows that remained after the filter has been applied

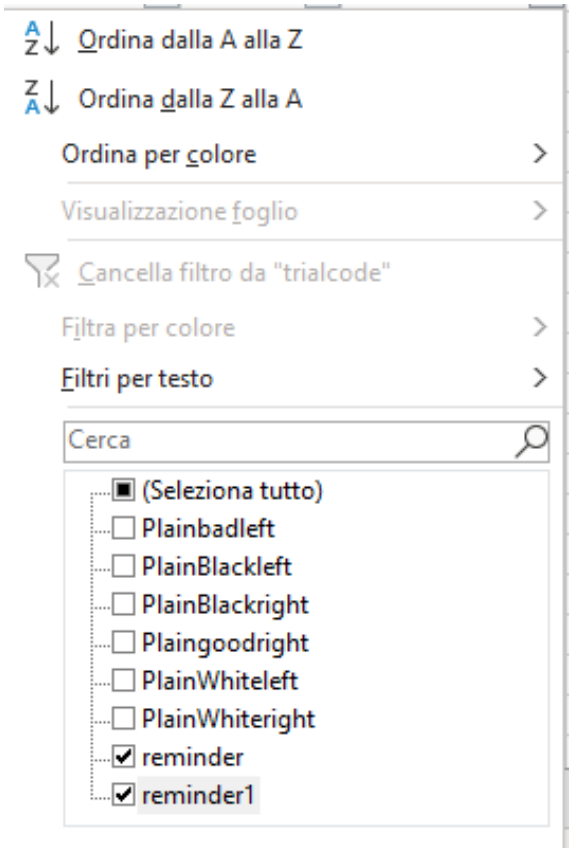
Don't panic when you see you have no more rows

Remove the filter from `blockcode` and it's all done. . . ish

trialcode

After selecting the blocks of interest, we still have some trials that are part of the experiment but of no interest for the *D* score computation (e.g., reminders, instructions)

Set a filter on the `trialcode` column and select the non-relevant trials:



`trialcode`

Select all the rows left after applying the filter

Delete them

(Again, don't panic)

Remove the filter from the `trialcode` column

Polish the data set

Remove any filter left

Remove all the unnecessary columns (`blocknum`, `trialnum`, `trialcode`, `response`, `stimulusitem1`)

Rename the remaining columns according to the shiny app instructions:

- **`subject`** → **`participant`**
- **`blockcode`** → **`block`**
- **`latency`** → **`latency`**
- **`correct`** → **`correct`**

The final look

	A	B	C	D	E
1	participant	block	correct	latency	
2	1	PracticeWhitebad	1	725	
3	1	PracticeWhitebad	1	1052	
4	1	PracticeWhitebad	1	1517	
5	1	PracticeWhitebad	1	767	
6	1	PracticeWhitebad	1	985	
7	1	PracticeWhitebad	1	708	
8	1	PracticeWhitebad	1	689	
9	1	PracticeWhitebad	1	719	
10	1	PracticeWhitebad	1	550	
11	1	PracticeWhitebad	1	1101	
12	1	PracticeWhitebad	1	918	
13	1	PracticeWhitebad	1	812	
14	1	PracticeWhitebad	1	717	
15	1	PracticeWhitebad	1	1028	
16	1	PracticeWhitebad	1	823	
17	1	PracticeWhitebad	1	843	
18	1	PracticeWhitebad	1	764	
19	1	PracticeWhitebad	1	651	
20	1	PracticeWhitebad	1	1076	

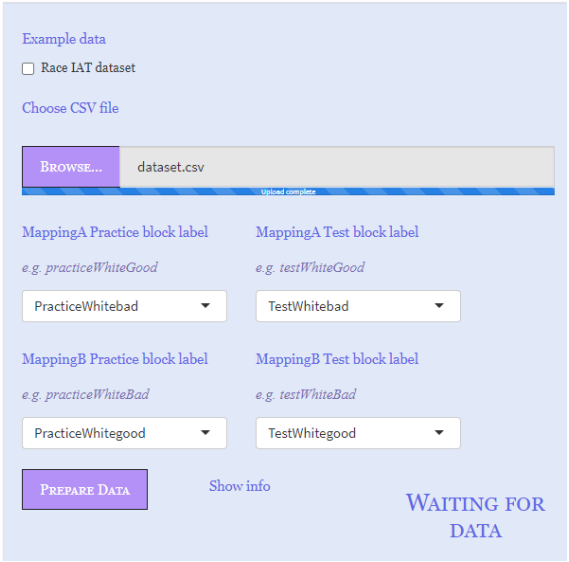
The file must be saved as `.csv` with "," specified as the column separator

It's super important that "," is set as the column separator because otherwise the app won't work and won't throw an error (because the maintainer thought "I'll add the error message tomorrow")

Upload the data set

Browse through your files, select the file you have just created. It will be automatically uploaded

Prepare the data set

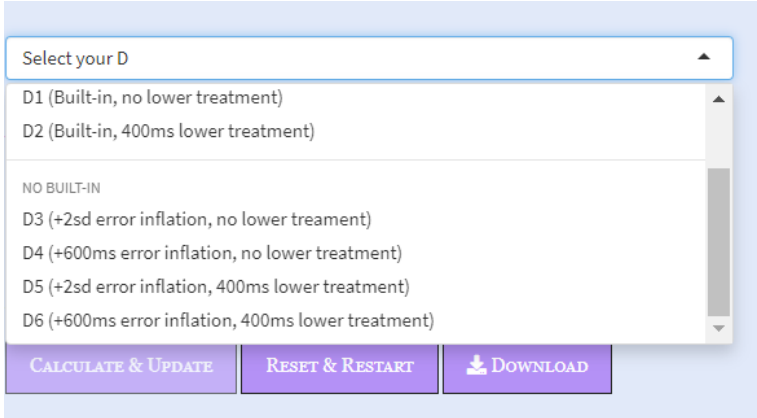


To change the mapping for the computation of the D score, just select the labels of the blocks from the drop down menu

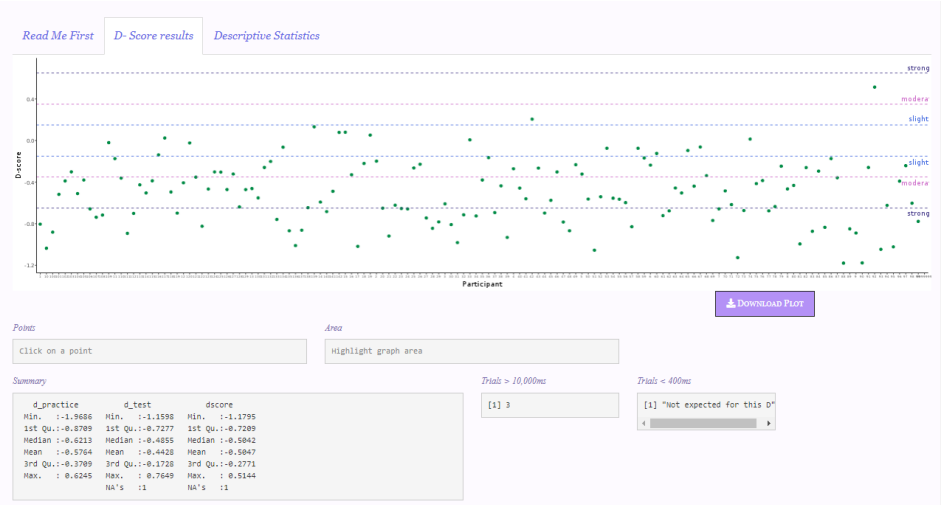
When you're all set just click "Prepare data" and wait for the "Data are ready" message to appear.

Select your *D*

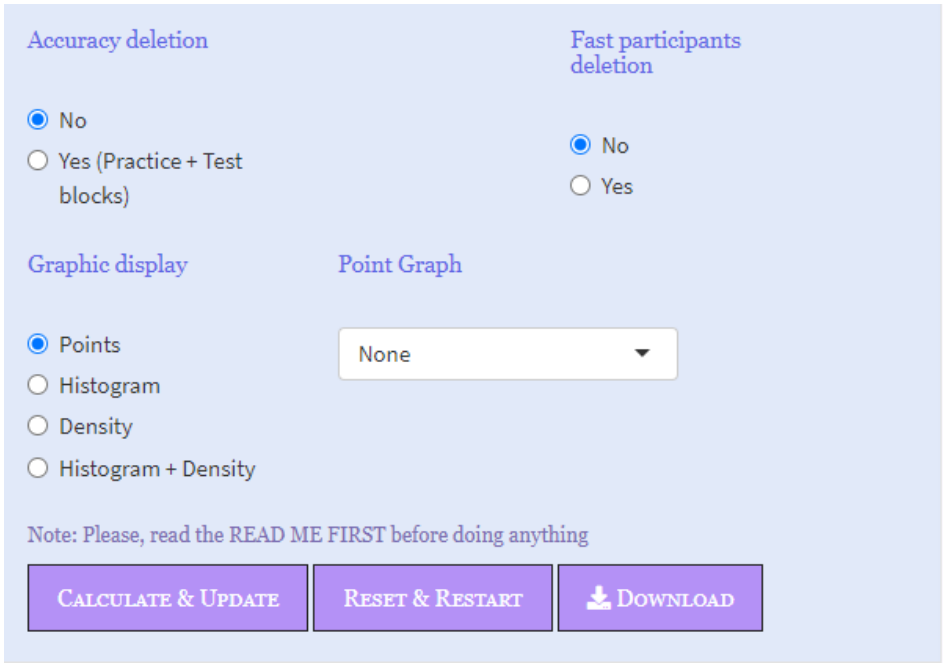
Just select the *D* score you want to compute from the drop down menu, click on "Calculate & Update", and it's all done! The *D* scores will shortly appear in the Results panel



(default) Results panel



Play with graphical representations and settings



Download

Once you're done, you can download the results in a `.csv` file (“,” is the column separator)

The name of the file contains the label of the last *D* score you have computed

For instance, if *D3* is the last algorithm you have computed, than the file name will be: `ShinyAPPDscore3.csv`

implicitMeasures

If you are an `R` user, `implicitMeasures` is made for you!

Adavantages

- It's super easy to use
- Well documented and the maintainer is always available to help you
- You can compute multiple *D* scores all at once
- Computes the score also for the SC-IAT
- You can compute the *D* score and run further analysis all in the same place

Disadvantages

- You have to be familiar with `R` to efficiently use `implicitMeasures`

Install & Upload

Install package:

```
> install.packages("implicitMeasures")
```

Upload the package so you can use its functions:

```
> library(implicitMeasures)
```

and now you're good to go

Import dataset

Again, we use the data set obtained from Inquisit (file .dat):

```
> data = read.table("IATdata.dat", header = TRUE, sep = "\t")
> head(data)
```

```
    date  time   build subject blocknum  blockcode trialnum        ti
1 121318 09:55 3.0.6.0       1        1   consenso        1         c
2 121318 09:55 3.0.6.0       1       19 WhiteBlack        1         i
3 121318 09:55 3.0.6.0       1       19 WhiteBlack        2  PlainWh
4 121318 09:55 3.0.6.0       1       19 WhiteBlack        4 PlainBla
5 121318 09:55 3.0.6.0       1       19 WhiteBlack        6  PlainWh
6 121318 09:55 3.0.6.0       1       19 WhiteBlack        8 PlainBla
                                                          response
1 Disponibile a partecipare alla ricerca nei termini sopra indicati
2                                                                57
3                                                                18
4                                                                23
5                                                                18
6                                                                23
```

blockcode

```
> table(data$blockcode)

          badgood         consenso       demografica  PracticeWhite
             5985              147              852              2
PracticeWhitegood     TestWhitebad     TestWhitegood        WhiteBl
             3003             5822             5863              3
    WhiteBlack2nd
             3003
```

We have a lot of stuff to get rid of. . . .

trialcode

```
> table(data$trialcode)

      consenso            edu            età      occupazio
           147            142            142            142
PlainBlackleft PlainBlackright  Plaingoodright  PlainWhiteleft Pla
          3575           3561           7125           3560
          pol1           pol2       reminder      reminder1
           142            142            857            285
```

Prepare data set

```
> data_clean = clean_iat(
+   data, # data set name
+   sbj_id = "subject", # column of sbj IDs
+   block_id = "blockcode", # column of the block labels
+   mapA_practice = "PracticeWhitegood",
+   mapA_test = "TestWhitegood",
+   mapB_practice = "PracticeWhitebad",
+   mapB_test = "TestWhitebad",
+   latency_id = "latency", # column with latency
+   accuracy_id = "correct", # column with accuracy
+   trial_id = "trialcode", # column with trial labels
+   trial_eliminate = c("reminder", "reminder1"), # trials to get rid of
+   demo_id = "blockcode", # column of the block labels
+   trial_demo = "demografica" # label of the demographic trials
+ )
```

```
> names(data_clean)

[1] "data_keep"      "data_eliminate" "demo"
```

- data_keep: the data set on which we compute the D scores (with class data.frame, iat_clean)
- data_eliminate: all the discarded trials
- demo: a data set containing all the demographic infos (we will see it later!)

```
> names(data_clean)

[1] "data_keep"      "data_eliminate" "demo"
```

- data_keep: the data set on which we compute the D scores (with class data.frame, iat_clean)
- data_eliminate: all the discarded trials
- demo: a data set containing all the demographic infos (we will see it later!)

If you export in `.csv` data_keep you can upload it to DScoreApp!

```
> write.table(data_clean[[1]], "cleanIAT.csv", sep = ",", row.
```

Compute *D* score

```
> iat = data_clean[[1]]
```

Use the `compute_iat()` function:

```
> d3 = compute_iat(iat, Dscore = "d3")
```

```
> head(d3[, 1:5])
```

```
  participant n_trial nslow10000 nfast400 nfast300
1           1     120          0     0.01        0
2           2     120          0     0.03        0
3           3     120          0     0.14        0
4           4     120          0     0.07        0
5           5     120          0     0.00        0
6           6     120          0     0.00        0
```

```
> head(d3[, 6:10])

  accuracy.practice_MappingA accuracy.practice_MappingB accuracy.tes
1                       1.00                       1.00
2                       1.00                       0.95
3                       0.95                       0.75
4                       1.00                       0.95
5                       0.95                       1.00
6                       0.95                       0.90
  accuracy.test_MappingB accuracy.MappingA
1                  0.950         0.9833333
2                  1.000         0.9833333
3                  0.900         0.9000000
4                  0.950         0.9666667
5                  0.925         0.9333333
6                  0.975         0.9166667
```

```
> head(d3[, 11:15])

  accuracy.MappingB RT_mean.MappingA RT_mean.MappingB mean_practice_
1         0.9666667          597.6649          738.7075
2         0.9833333          598.3324          649.2085
3         0.8500000          575.2006          721.7637
4         0.9500000          606.2957          645.4930
5         0.9500000          849.8184         1011.9773
6         0.9500000          914.6978          981.7482
  mean_test_MappingA
1           585.7973
2           589.5736
3           585.2044
4           618.3685
5           695.3284
6           911.1501
```

```
> head(d3[, 16:19])

  mean_practice_MappingB mean_test_MappingB d_practice_d3   d_test_d
1                851.5500           682.2862    1.00062841  0.6082558
2                840.5754           553.5250    0.62762704 -0.2318176
3                973.0089           596.1411    1.16298569  0.0550429
4                746.2660           595.1065    0.70572685 -0.1628654
5               1134.5500           950.6909   -0.04599615  0.6910868
6               1152.3929           896.4259    0.50668795 -0.0342553
```

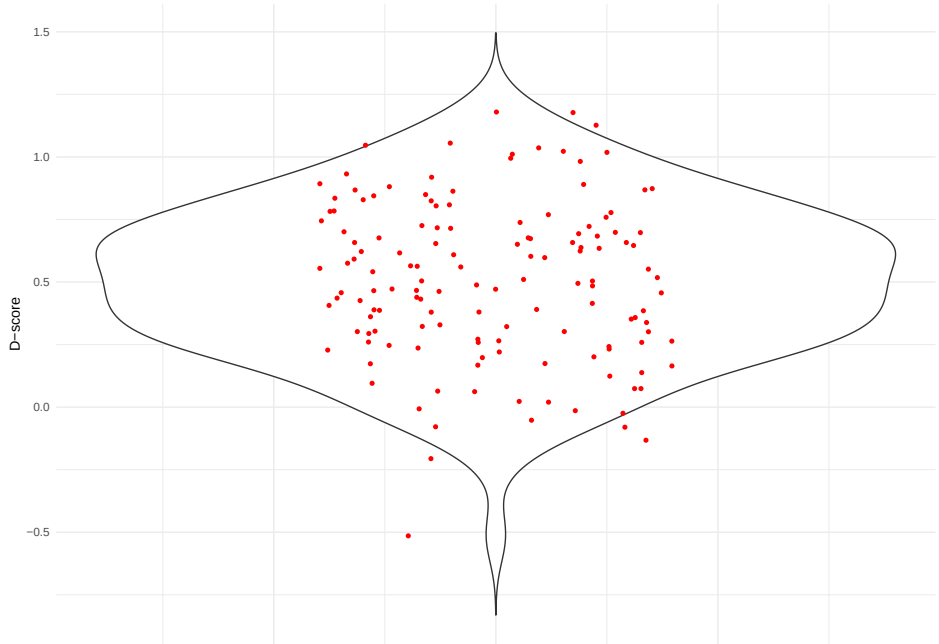
```
> head(d3[, 20:21])

  dscore_d3       cond_ord
1 0.8044421 MappingB_First
2 0.1979047 MappingA_First
3 0.6090143 MappingB_First
4 0.2714307 MappingA_First
5 0.3225454 MappingA_First
6 0.2362163 MappingB_First
```

```
> head(d3[, 22:23])
```

	legendMappingA	legendMapp
1	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite
2	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite
3	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite
4	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite
5	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite
6	PracticeWhitegood_and_TestWhitegood	PracticeWhitebad_and_TestWhite

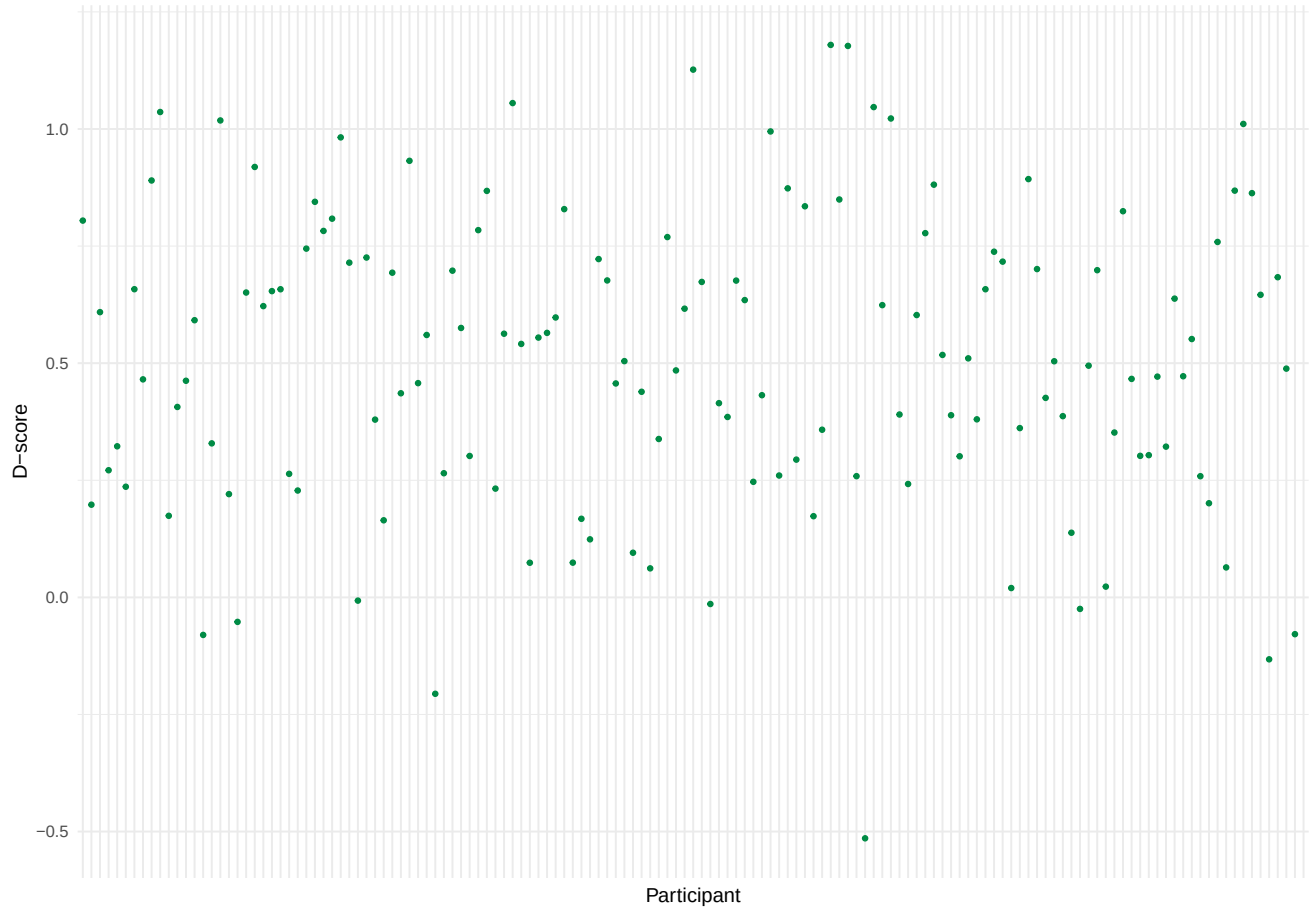
Some plots!

```
> d_density(d3, graph = "violin")
```



Other plots

```
> d_point(d3, x_values = FALSE)
```



Multiple *D* scores at once

```
> dscores = multi_dscore(iat, ds = "error-inflation")
```

Careful!

```
> names(dscores)
```

```
[1] "dscores" "graph"
```

```
> head(dscores[[1]])

  participant dscore_d3 dscore_d4 dscore_d5  dscore_d6
1           1 0.8044421 0.7664824 0.7902660 0.75162963
2           2 0.1979047 0.1807094 0.1081013 0.09710327
3           3 0.6090143 0.5433544 0.6836311 0.65357652
4           4 0.2714307 0.2849834 0.4108129 0.43049456
5           5 0.3225454 0.3177612 0.3225454 0.31776123
6           6 0.2362163 0.2714998 0.2362163 0.27149977
```

```
> dscores[[2]]
```


1.5
1.0
0.5
0.0
−0.5
D−scores
dscore_d3
dscore_d4
dscore_d5
dscore_d6
D Error Inflation

Did we forget about demo...?

```
> demo_raw = data_clean[[3]]
> str(demo_raw)

'data.frame':   852 obs. of  12 variables:
 $ date          : int  121318 121318 121318 121318 121318 121318 121
 $ time          : chr  "09:55" "09:55" "09:55" "09:55" ...
 $ build         : chr  "3.0.6.0" "3.0.6.0" "3.0.6.0" "3.0.6.0" ...
 $ participant   : int  1 1 1 1 1 1 2 2 2 2 ...
 $ blocknum      : int  57 57 57 57 57 57 57 57 57 57 ...
 $ blockcode     : chr  "demografica" "demografica" "demografica" "de
 $ trialnum      : int  1 1 1 1 4 4 1 1 1 1 ...
 $ trialcode     : chr  "sesso" "età" "occupazio" "edu" ...
 $ response      : chr  "Maschio" "21" "stud" "sup" ...
 $ correct       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ latency       : int  19185 19185 19185 19185 28866 28866 24586 245
 $ stimulusitem1: chr  "0" "0" "0" "0" ...
....
```

Let's start by keeping only the column of interest:

```
> demo_raw = demo_raw[, c("participant", "trialcode", "respons
> str(demo_raw)

'data.frame':   852 obs. of  3 variables:
 $ participant: int  1 1 1 1 1 1 2 2 2 2 ...
 $ trialcode  : chr  "sesso" "età" "occupazio" "edu" ...
 $ response   : chr  "Maschio" "21" "stud" "sup" ...
```

Reshape the data set

```
> demo <- reshape(demo_raw, timevar = "trialcode", idvar = "pa
> str(demo)

'data.frame':   142 obs. of  7 variables:
 $ participant        : int  1 2 3 4 5 6 7 8 9 10 ...
 $ response.sesso     : chr  "Maschio" "Maschio" "Femmina" "Fem
 $ response.età       : chr  "21" "31" "21" "21" ...
 $ response.occupazio: chr  "stud" "stud" "stud" "stud" ...
 $ response.edu       : chr  "sup" "magistrale" "sup" "sup" ...
 $ response.pol1      : chr  "3" "3" "3" "2" ...
 $ response.pol2      : chr  "2" "5" "2" "3" ...
....
```

Polish the data set

```
> colnames(demo) <- gsub("response.", "", colnames(demo))
> demo[, c(3, 6:7)] <- apply(demo[, c(3, 6:7)], 2, as.integer)
> str(demo)

'data.frame':   142 obs. of  7 variables:
 $ participant: int  1 2 3 4 5 6 7 8 9 10 ...
 $ sesso      : chr  "Maschio" "Maschio" "Femmina" "Femmina" ...
 $ età        : int  21 31 21 21 21 20 20 19 20 20 ...
 $ occupazio  : chr  "stud" "stud" "stud" "stud" ...
 $ edu        : chr  "sup" "magistrale" "sup" "sup" ...
 $ pol1       : int  3 3 3 2 4 3 2 3 3 3 ...
 $ pol2       : int  2 5 2 3 4 2 3 2 1 2 ...
....
```

Merge demo with d3

```
> d3complete = merge(d3, demo, by = "participant")
> str(d3complete[17:29])

'data.frame':   142 obs. of  13 variables:
 $ mean_test_MappingB: num  682 554 596 595 951 ...
 $ d_practice_d3     : num  1.001 0.628 1.163 0.706 -0.046 ...
 $ d_test_d3         : num  0.608 -0.232 0.055 -0.163 0.691 ...
 $ dscore_d3         : num  0.804 0.198 0.609 0.271 0.323 ...
 $ cond_ord          : chr  "MappingB_First" "MappingA_First" "Mapp
 $ legendMappingA    : chr  "PracticeWhitegood_and_TestWhitegood" "
 $ legendMappingB    : chr  "PracticeWhitebad_and_TestWhitebad" "Pra
 $ sesso             : chr  "Maschio" "Maschio" "Femmina" "Femmina"
 $ età               : int  21 31 21 21 21 20 20 19 20 20 ...
 $ occupazio         : chr  "stud" "stud" "stud" "stud" ...
 $ edu               : chr  "sup" "magistrale" "sup" "sup" ...
 $ pol1              : int  3 3 3 2 4 3 2 3 3 3 ...
 $ pol2              : int  2 5 2 3 4 2 3 2 1 2 ...
```

Compute correlation

```
> correlations <- data.frame(cor(d3complete[, c("dscore_d3", "pol1"
> correlations <- round(correlations, 2)
> correlations[upper.tri(correlations, diag = TRUE)] <- ""
```

Results are in Table 1:

Table 1: Race IAT correlations

	dscore_d3	pol1	pol2
dscore_d3			
pol1	0		
pol2	0.12	-0.34	

Supplementary material

DScoreApp:

Epifania, O. M., Anselmi, P., & Robusto, E. (2020). Dscoreapp: A shiny web application for the computation of the implicit association test D score. Frontiers in Psychology, 10, 2938. doi: 10.3389/fpsyg.2019.02938

`implicitMeasures`:

Epifania, O. M., Anselmi, P., & Robusto, E. (2020). Implicit measures with reproducible results: The implicitmeasures package. Journal of Open Source Software, 5(52), 2394. doi: 10.21105/joss.02394