Decoding Trial Data, Scoring Tasks, and Widening

*Thomas Pronk*

[*tpronk@uva.nl*](mailto:tpronk@uva.nl)

*Version 1.2, 2016-04-16*

This tutorial assumes you have installed R/RStudio and the SANDRA library, which you have already done if you have followed the steps of Tutorial *1. Installing SANDRA.docx*.

# Introduction

This is a guide on how to process data from reaction time tasks via SANDRA. Processing data is done in three steps, as illustrated in Figure 1. These steps are:

* **t.1 - Decoding task data encoded in a LOTUS results file into trial data and metadata**
* **t.2 - Calculating scores from trial data**
* **t.3 - Merging scores together into a single wide file**



**Figure 1.** The workflow of decoding trial data, scoring tasks, and widening. Blue boxes represent SANDRA scripts and red boxes represent data files. All data files after t.1 are tab-separated by default.

# ****Part 1. Setup the pre-provided SANDRA analysis framework****

The SANDRA GitHub repository provides an analysis framework with some pre-made data and processing scripts that can easily be adjusted to your needs.

1. Download the whole repo in a ZIP via the link below. See “Download ZIP” button in the top right corner of the screen. <https://github.com/tpronk/SANDRA>
2. Unzip the file. The folder *framework\_demos* contains a SANDRA Analysis Framework, together with the processing scripts, and some example data to test them on.
3. Move the contents of *framework\_demos* to a comfortable location on your computer, then run “Install SANDRA (from GitHub).R” to set up the framework for that location. This is similar to what you did in the tutorial Installing SANDRA, Step 2, except that in this case the analysis framework already contains data and scripts.

If everything went well, you’ll see some messages saying that various directories already exist, and finally the message:

sandra::FrameworkFileIO. Succesfully constructed FileIO

# ****Part 2. Test the processing script with the example data****

The *scripts* folder contains a collection of scripts for processing your data. In general, you’ll execute a t.1, t.2, and t.3 script in order. Below is a description of what each t.1, t.2, and t.3 script does, what they produce, and how they can be configured.

* t.1 decodes tasks data from a LOTUS results file into trial data and metadata, the latter of which contains participant parameters and some reporting variables. There are scripts available for JASMIN1 data (for all JASMIN tasks until the present) and SPRIF data (for Flash tasks).
  + It produces a *trialdata* file (one file per task; one row per trial) and *metadata* file (one file per dataset; one row per participation). The *trialdata* and *metadata* files can be joined together on the *set\_id­* variable.
  + You can configure *fileSource* (which file to decode) and *participationID* (which columns identify a participation).
* t.2 calculates scores (and/or split-halve reliabilities) from trial data. There are scripts available for d-scores, and for aggregations such as “difference of medians/means for correct responses.
  + In the case of scores, the script produces a *scores* file (one per task; one row per participation), but in the case of split-halve reliabilities, it only prints a single value to the R Console.
  + You can configure *fileSource* and *scorings* (how to score a task). Type **?calculateScores**into the R console to get more information about how to setup the task scoring.
* t.3 merges scores across tasks and sessions together into a single ‘wide’, with one row per participant and columns postfixed by task and session.
  + It produces a *joined* file (one file per dataset, one row per participant)
  + You can configure *fileSource*, participantID (which column identifies one participant), *sessionID* (which column identifies a session), and *tasks* (which task data to combine). Finally, you can drop artefacts before joining the files together via the *dropArtefacts* function.

By default, the scripts are set up to process a demo dataset named “jasmin1\_data.csv” located in the *original* folder. Run one t.1, t.2, and t.3 script in order to test if the analysis framework is set up correctly. If no errors occurred, then a set of data files should be produced in the *interim* folder with names such as “jasmin1\_data.trialdata.vpt” and “jasmin1\_data.scores.vpt”. Check “date modified” to find out whether these files were actually just created.

# ****Part**** 3. Download a LOTUS Results File

Time to download your task data into a results file. If you are downloading JASMIN1 data, follow the instructions in 3a; if you are downloading SPRIF data, follow the instructions in 3b.

## ****Part**** 3.a. JASMIN1 Data

1. Login to LOTUS.
2. Go to *Your project 🡪 View participants 🡪 Participants 🡪 Results*.
3. Click *Export* in the top right of the screen. At the export screen:
   1. Make sure to select any participant parameters you are interested in (such as *username* or *age)*.
   2. At Separator, remove the ‘;’ and enter ‘\t’ instead.
   3. At Escape, remove the ‘\’ and just leave that field empty.
   4. Click OK.
4. Unzip the downloaded file and put it in the *original* folder.
5. Be sure to give your unzipped results file an informative name, such as *pretest\_alcohol.csv*. Note that by default, Windows Explorer does not display the *extension* of a filename (the part after the dot, which is *csv* in the example. In that case, you don’t need to enter the extension when renaming the file; just name it *pretest\_alcohol*. However, when adjusting *fileSource* in the processing script, this extension needs to be specified. To summarize, in Explorer your file could be displayed as *pretest\_alcohol*, but in the scripts you need to refer to this file as *pretest\_alcohol.csv*.

## ****Part**** 3.b. SPRIF Data

1. Login to LOTUS.
2. Go to *Your project 🡪 View participants 🡪 Participants 🡪 Results*.
3. At the *Result name* drop-down menu at top of the screen, select *trial*
4. Click *Export* in the top right of the screen. At the export screen:
   1. Make sure to select any participant parameters you are interested in (such as *username* or *age)*.
   2. Click OK.
5. Unzip the downloaded file and put it in the *original* folder.
6. Be sure to give your unzipped results file an informative name, such as *pretest\_alcohol.csv*. Note that by default, Windows Explorer does not display the *extension* of a filename (the part after the dot, which is *csv* in the example. In that case, you don’t need to enter the extension when renaming the file; just name it *pretest\_alcohol*. However, when adjusting *fileSource* in the processing script, this extension needs to be specified. To summarize, in Explorer your file could be displayed as *pretest\_alcohol*, but in the scripts you need to refer to this file as *pretest\_alcohol.csv*.

# ****Part 4. Process your own data****

Now configure the right t.1, t.2, and t.3 scripts and process your data.

* At the very least, in every script, you’ll need to change *fileSource* to point to the results file you just downloaded. See “Step 2” and the comments inside the processing scripts for more information about how to configure each script.
* Note that with big datasets, t.1 may take a while.
* Don’t forget to set up scorings for each of the tasks encoded in your data
* The manual *TP - CBM Tasks JASMIN Configuration.docx* provides more information about the meaning of all the variables in the trial data. You can find this manual in ADAPT Shared/Documentation/CBM Tasks

# ****Appendix 1. Metadata variables for JASMIN data****

In the case of JASMIN data, SANDRA will add a couple of variables to the metadata that provide some additional information about trial data processing. These variables are also added to the *scores* and *joined* files. Table 1 explains the meaning of these variables.

**Table 1.** Explanation about JASMIN metadata variables

|  |  |
| --- | --- |
| **Folder** | **Explanation** |
| run\_from | Earliest RunID in results file with data for this participation |
| run\_to | Latest RunID results file with data for this participation |
| lotus\_says | Tells you how the task ended. There is only trialdata produced if the task completed successfully. Possible values:   * **task\_done**. Task completed successfully * **task\_start**. Task was restarted before it completed * **task\_error**. Task reported that an error occurred |
| event\_count | The number of events logged during the task, may vary depending on participant behavior. |
| sequence\_report | If this variable is not empty, then the trial data of this participation are suspicious. Possible values:   * **inconsistent.** Events with the same sequence number but different data. This can happen if multiple participants are taking part in a task using the same LOTUS account. * **missing.** Sequence numbers missing. This indicates that certain task data was not received by LOTUS. * **negtime.** Client time decreased with successive sequence numbers. This indicates that the participants’ computer had an unreliable clock. * **invalid.** Task produced invalid (or no) data. This can happen when the participant did not complete any trials. |
| taskName | The type of task belonging to this row in metadata. |

# ****Appendix 2. Metadata variables for SPRIF data****

In the case of SPRIF data, SANDRA will add a couple of variables to the metadata that provide some additional information about trial data processing. These variables are also added to the *scores* and *joined* files. Table 2 explains the meaning of these variables.

**Table 2.** Explanation about SPRIF metadata variables

|  |  |
| --- | --- |
| **Folder** | **Explanation** |
| sequence\_report | If this variable is not empty, then the trial data of this participation are suspicious. Possible values:   * **incomplete.** The number of values in SPRIF data was not a whole multiple of the number of elements in (sprifVars + 1). This indicates that trials in the task produced different numbers of variables. * **norowsep.** One of the rows of SPRIF data did not start with a SPRIF row separator. This indicates that trials in the task produced different numbers of variables. * **rowsepinvar.** One of the SPRIF variables contained a SPRIF row separator. This indicates that trials in the task produced different numbers of variables. * **invalid.** Task produced invalid (or no) data. This can happen when the participant did not complete any trials. |
| taskName | The type of task belonging to this row in metadata. |

# ****Acknowledgements****

My gratitude goes to many researchers that provided help. In particular, I would like to thank Joeri van Wijngaarden for co-developing tutorial materials, and Marilisa Boffo and Ruby Smits for testing tutorials and providing feedback on usability.