TimeStudio process manual – Pupil analysis

Author: Nils Olofsson and Pär Nyström
Last revision: 2013-04-15

Index

1. Purpose

This manual contains step by step instructions of how to analyse pupil data in the TimeStudio framework. The manual is written for novel users of TimeStudio.

This manual assumes that you have downloaded the latest TimeStudio core and the data files mentioned below. If you do not know how to start TimeStudio, please double check that you have followed the steps described in http://TimeStudioproject.com/getting-started

2. Definitions

Buttons that need to be activated in order to execute a command or fields within the workspace are indicated with quotation marks in the text below. Plugin names are marked with bold. Letters in red circles in figures and text indicate where in TimeStudio a particular action takes place.

3. Dataset

The dataset includes recordings of eye movements recorded with a Tobii 1750 eye tracker and exported as text files from ClearView (other formats are also supported, for details see Appendix B). In this study five participants were asked to multiply two digits that appear one after the other on a screen. Using this paradigm Ahern and Betty (1979) demonstrated that the pupil dilates more following difficult than easy multiplications (see **Fel! Hittar inte referenskälla.**). These findings were interpreted in support the notion that pupil dilations are mediated by cognitive load (for alternative sources of pupil dilation in humans see Laeng, Sirois, and Gredebäck, 2012).

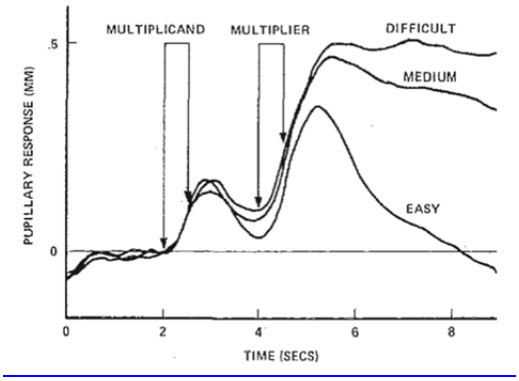

Figure 1. Pupil size as a function of processing load during a multiplication task (adapted from Ahern & Betty, 1974).

The files needed to follow this manual are:

pupil_subj1.txt
pupil_subj2.txt
pupil_subj3.txt
pupil_subj4.txt
pupil_subj5.txt

Each file is a recording of one test subject and the files can be downloaded from

http://www.TimeStudioproject.com/manuals/pupil_manual_datafiles.zip

4. Overview

This tutorial demonstrates how TimeStudio can be used to analyse pupil dilation data using the example specified in section 2. In order to complete the analysis a series of operations need to be performed in TimeStudio.

- Launch MATLAB and TimeStudio
- Add data and define participants
- Read added data files
- Change names of events in dataset
- Remove NaN-values
- Apply average filter
- Statistical extraction and visualization
- View results

When this tutorial is complete the workflow and results windows should look identical to Figure 2, replicating the original findings by Ahern and Beatty (1979).

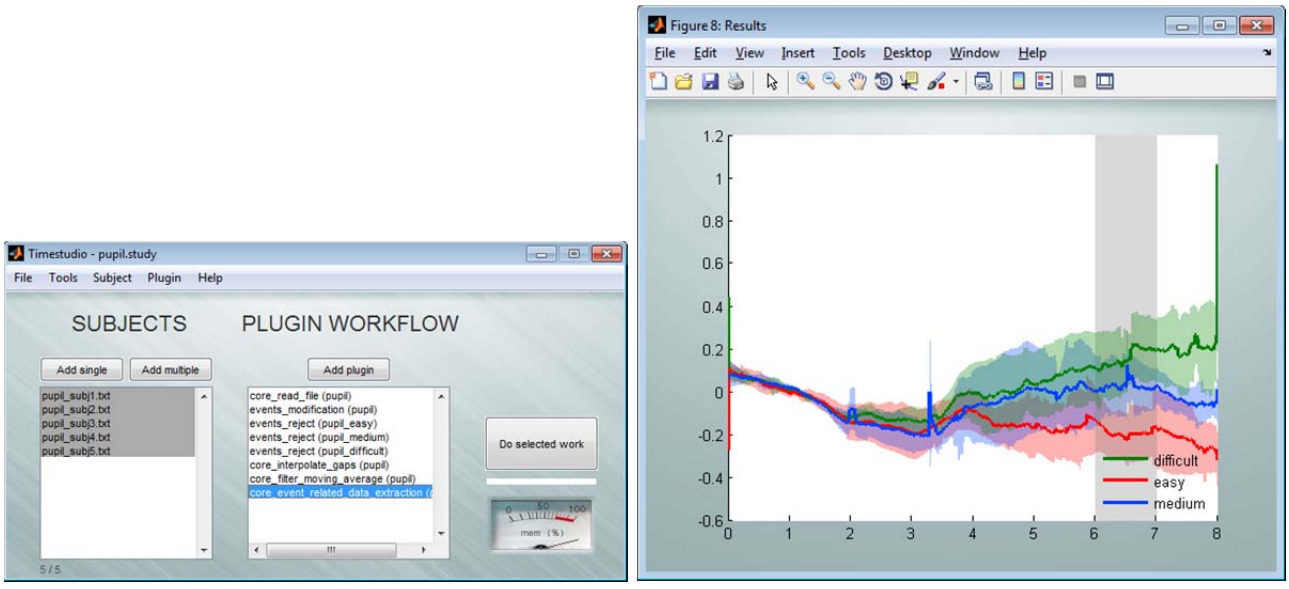

Figure 2. The final state of the workflow (left) and a graph showing the average pupil size per condition (right).

5. Step-by-step instructions

1.1 Launch MATLAB and TimeStudio

1. Start by launching MATLAB.

2. Either add the TimeStudio folder to the MATLAB path, or move into the TimeStudio folder.

3. Start TimeStudio by typing ‘TimeStudio’ in the MATLAB command window and press enter.

When TimeStudio starts the main window is displayed as seen in

Figure 3.
The empty field to the left (**Fel! Hittar inte referenskälla.**A) is called “subjects” or “list of subjects” and the empty field to the right (**Fel! Hittar inte referenskälla.**B) is called “plugin workflow”.

1.2 Add data and define participants

First we need to add the data files to the workflow.

Press the “Add multiple subjects” button (

1. Figure 3C) in the main TimeStudio window. This will open a file browser window.

2. Locate the five data files that belong to this demonstration, they should be named “pupil_subjx.txt” with x being 1 to 5.

3. In the pop-up file browser window (Figure 4), mark all five files and press the “Open”-button (Figure 4A) to add all five files and create an individual subject for each file.

This should add five subjects to the subject list (Figure 5A).

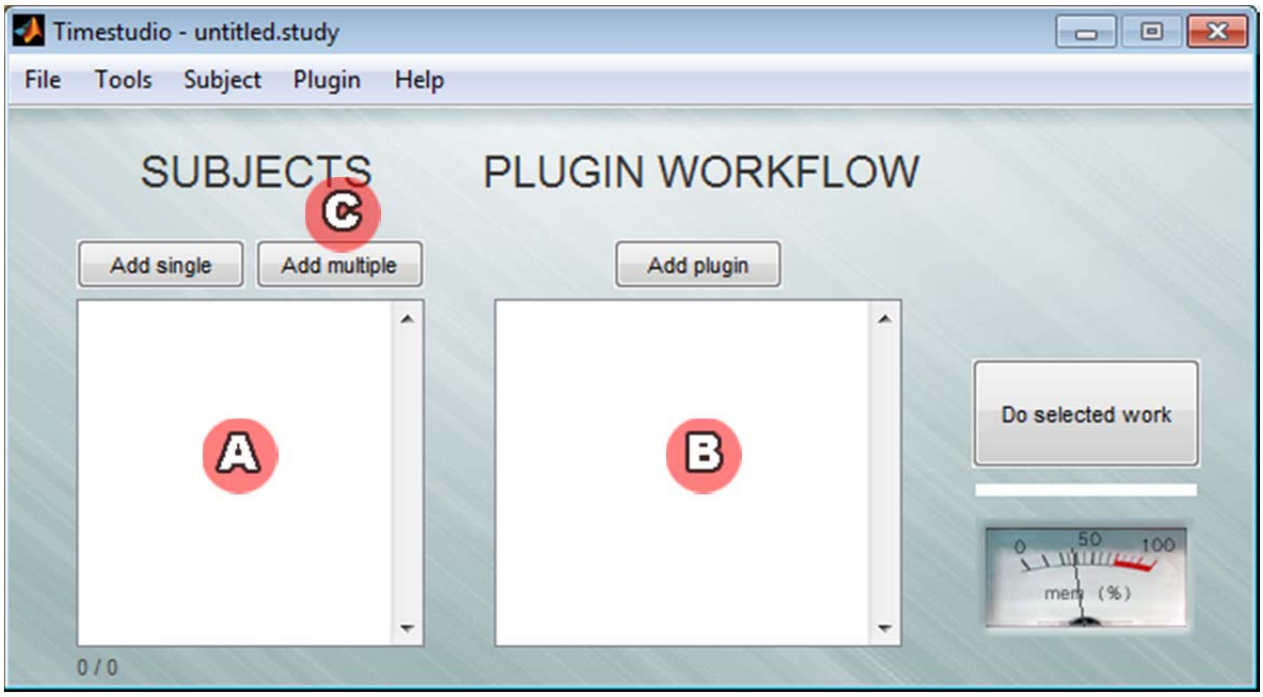

Figure 3. TimeStudio immediately after start-up.

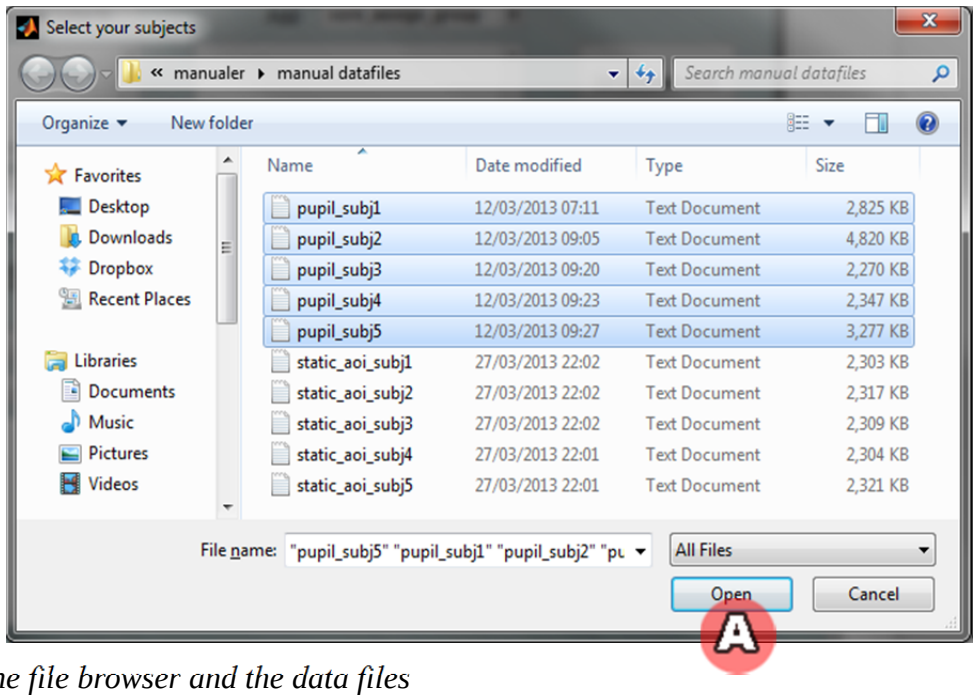

Figure 4. The file browser and the data files

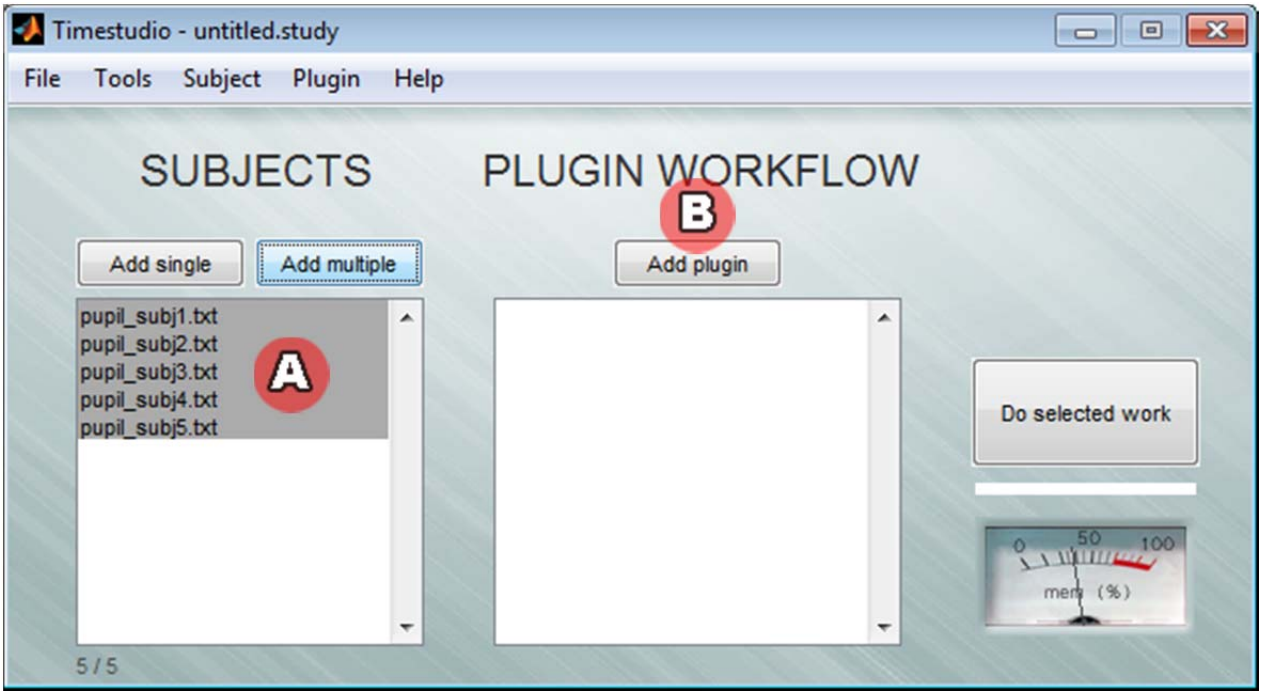

Figure 5. The TimeStudio main window after files has been added.

1.3 Read added data files

After we have created a list of subjects with associated data files we will need to read the content of those files into Matlab memory. This is done by adding a plugin called **core_read_file** to the study.

1. Press the “Add plugin” button (Figure 5B) in the TimeStudio main window to bring up the pop up menu for selecting and adding a plugin to the plugin workflow. Hover to “core” and select **core_read_file** from the pop-out menu. This should bring up the window seen in Figure 6.

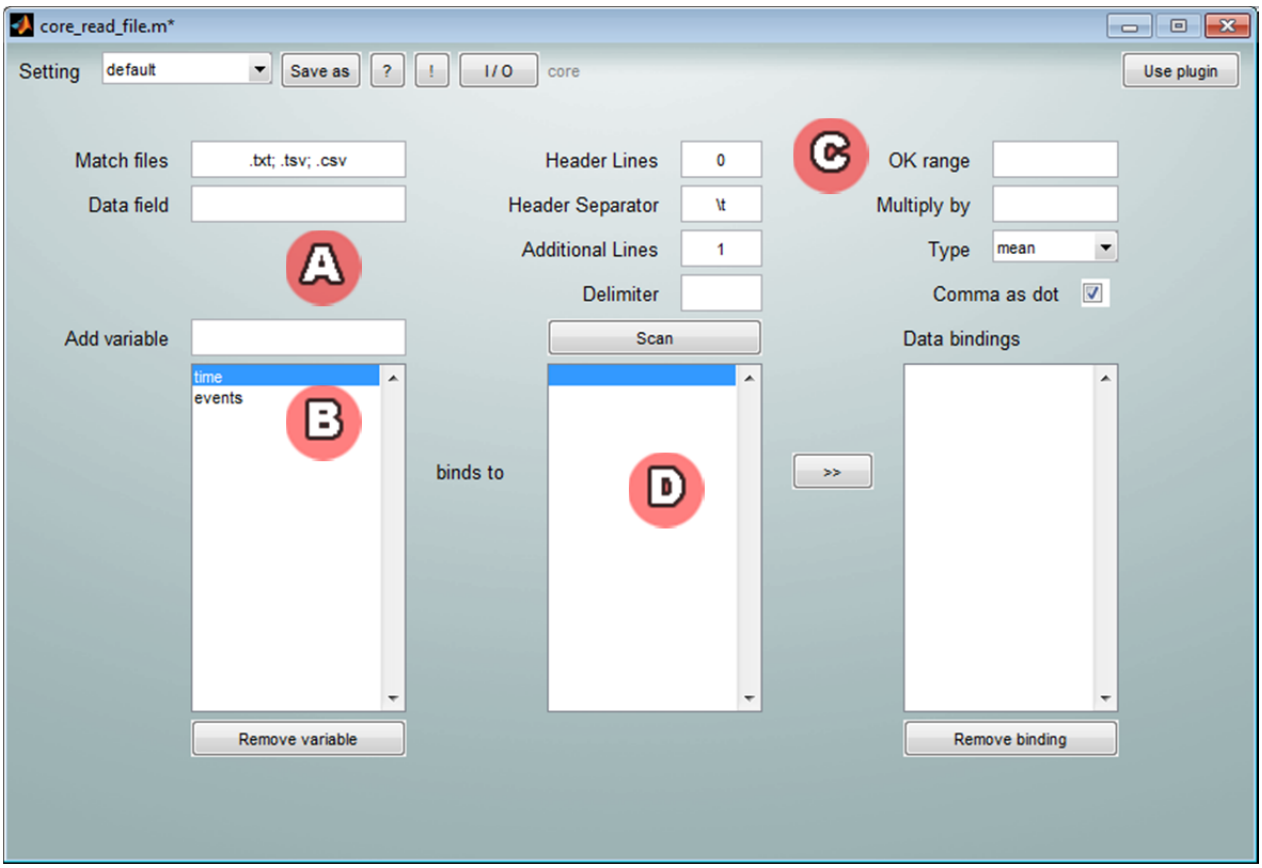

Figure 6. The core_read_file window.

2. Start with adding a data field. Above A, in Figure 6, enter ‘eyetracking’.

3. The next step is to add variables. They are entered by typing the variable name in the “Add variable”-box that is below the A, and pressing enter. In this way enter the variables ‘x’,’ y’, and ‘pupil’, one by one.

As can be seen in the listbox B, there are two predefined variables – ‘events’ and ‘time’. These are used and required by almost all plugins and are therefore added by default.

4. After this, the number of header lines in the files to be read should be entered in the “Header Lines”-box (C in Figure 6). In our case, enter ‘18’ in this box, and the properties (column names, that is) from our input files will be listed in the listbox at D in Figure 6.

5. This enables us to bind our variables to different data columns in the input files. Select ‘x’ in the variable listbox (B in Figure 6), then select ‘GazepointX’ in the properties listbox (D) and finally press the “>>”-button to the left of D the figure. This will create a binding that will be displayed in the “Data bindings”-listbox.

The results so far can be seen in Figure 7.

6. Continue with binding the variable ‘y’ to ‘GazepointY’ in the same way.

7. ‘pupil’ should be bound with the columns ‘Pupil (R)’ and ‘Pupil (L)’. Select both of these and do as before, but observe the “Type”-box next to A in Figure 7. The value here should be ‘mean’. This means that ‘pupil’ will be calculated as the mean value of ‘Pupil (R)’ and

‘Pupil (L)’.

8. Next up is to bind ‘time’ to ‘Timestamp’. This is done in the same way as for the other variables except that ‘0.001’ should be entered into the “Multiply by”-box (next to A in Figure 7). This means that the values of Timestamp are multiplied by 0.001, which gives is the time in seconds in this example.

9. As the final binding we should bind ‘events’ with ‘Description’. Now the “Type” should be set to ‘event’.

10. The final steps are to set the acceptable range for some of the variables. Select ‘x’ in the variable-listbox (at B in Figure 7), and then enter ‘0 1024’ in the “OK range”-box (close to A in Figure 7).

11. Next, select ‘y’ and set the ok range to ‘0 768’, and finish by selecting ‘pupil’ and setting the range to ‘2 10’.

Now that we have made all the settings required to read our data files we need to save them before using them.

12. Press “Save as” (above C in Figure 7). In the pop-up window enter some sensible name (e.g. ‘pupil’), then press “Save”.

13. Press “Use plugin” (at D in Figure 7) to close the settings window and add this plugin with these settings to the plugin workflow.

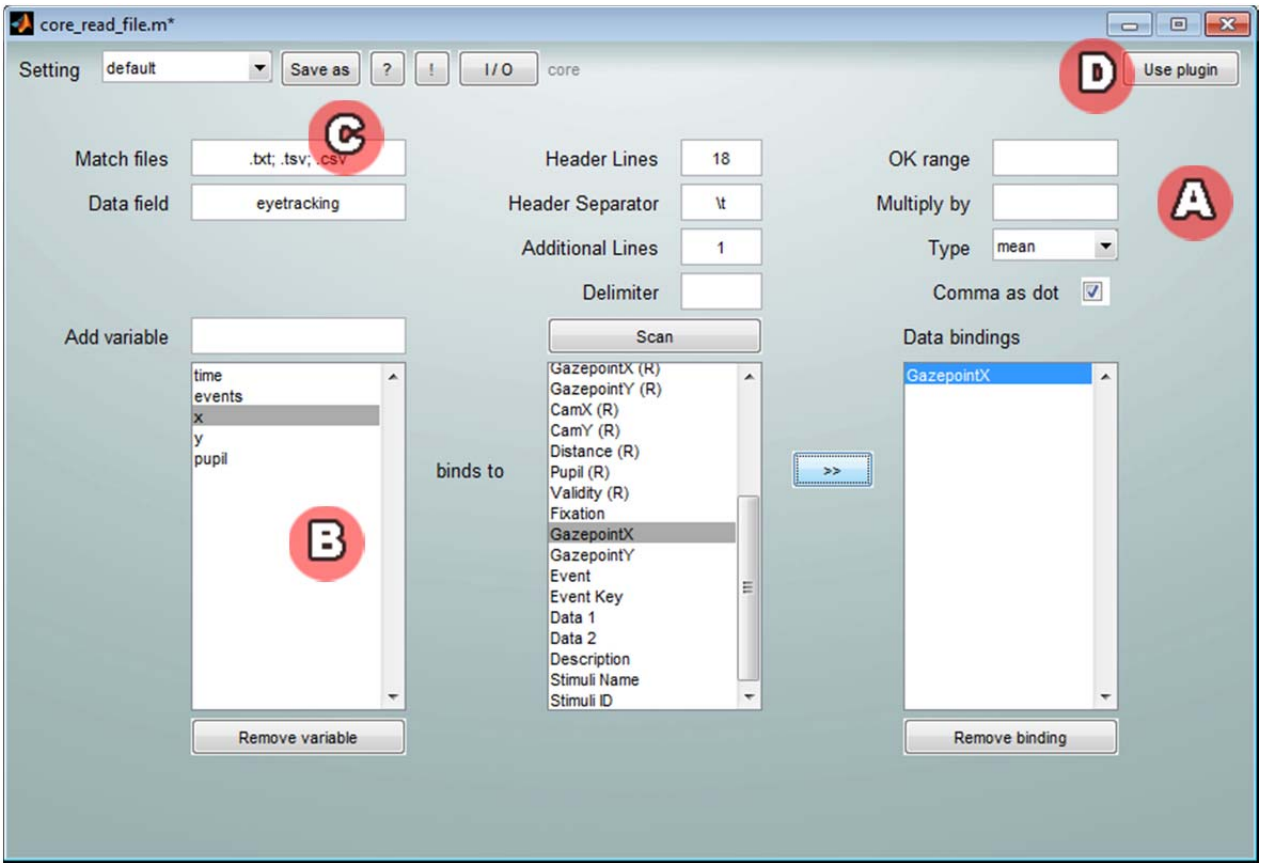

Figure 7. Adding bindings in core_read_file.

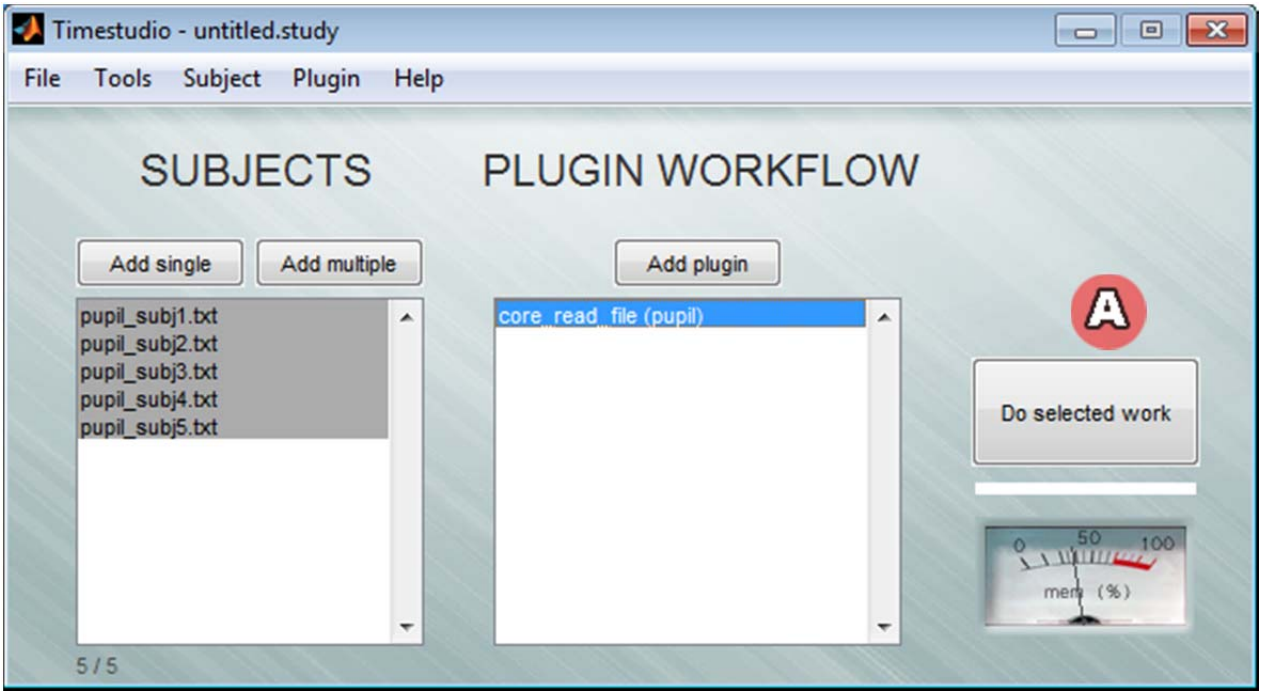

Figure 8. The TimeStudio main window after the plugin for file reading has been added.

Back in the main window:

1. Select all five subjects in the subject field, and select **core_read_file** in the plugin workflow (as can be seen in Figure 8).

2. Press the "Do selected work" button (Figure 8A) to read the data from the added files into the subjects.

While TimeStudio is processing a wait bar appears, and additional information is printed in the MATLAB command window.

1.4 Change names of events in dataset

Now we need to change the names of the events in the dataset so that events of the same kind are named the same.

Events which correspond to the subject performing a task classified as "difficult" should be named "difficult" etc. The plugin **events_modification** can be used to modify events by changing event values. All events have a name and a timestamp, and we will use the plugin to modify the name of some events but not the timestamp (other parameters may be assigned to any event by using this plugin, but we don't need that in this particular example. For more information on this see the help-file for the plugin). The plugin uses a list of rows and columns to modify the values. The plugin finds events which has a value in the first column and changes or adds the values in all other columns. This plugin can be used in many situations, but a common way to work with the plugin is described below.

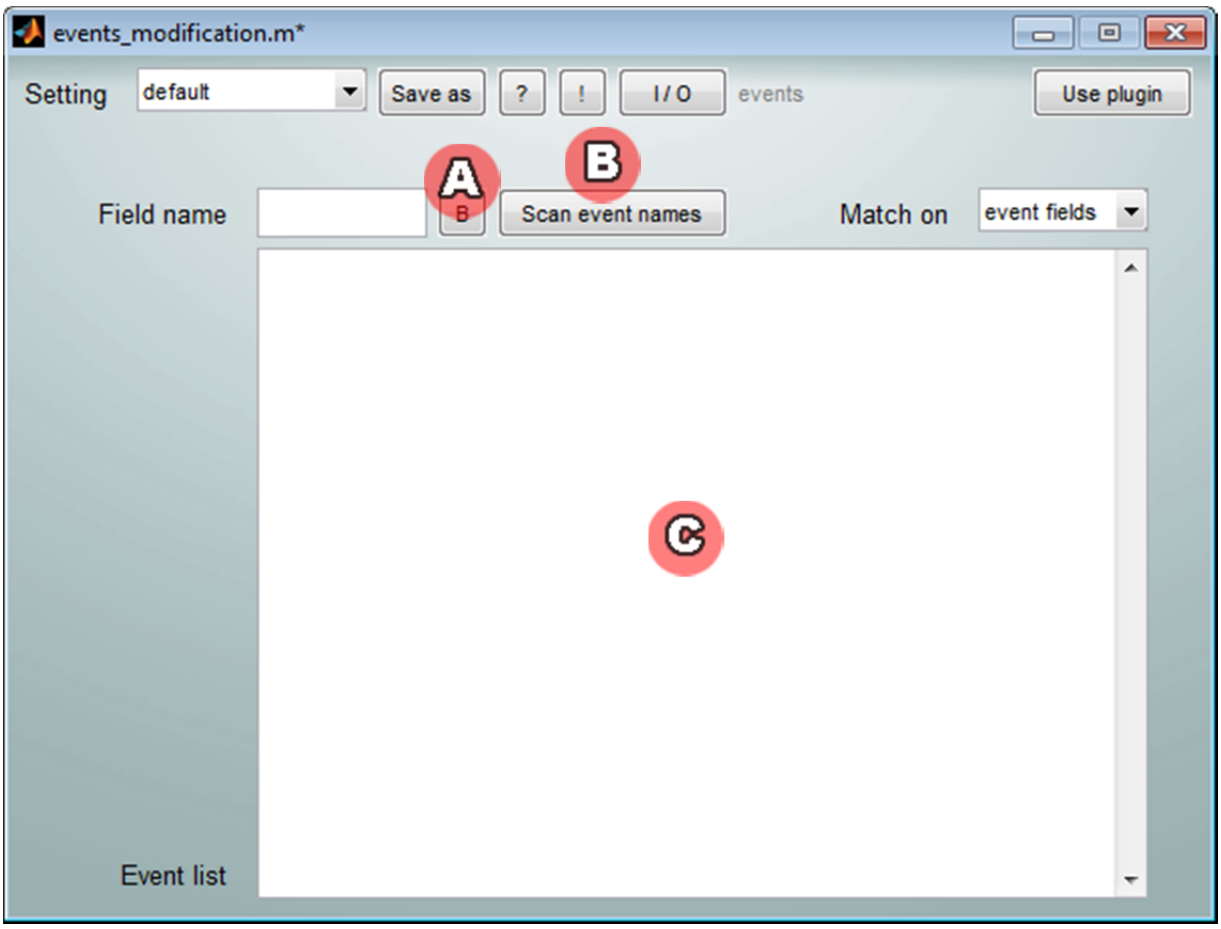

Figure 9. The empty setting window for the events_modification plugin.

1. As before, press “Add plugin” from the main menu to bring up the pop up for selecting plugins, press “events” and select **events_modification**. This will open up the settings window for the plugin, see Figure 9.

2. Browse (Figure 9A) for the name of the data field to use. The only field visible should be ‘eyetracking’. Select ‘eyetracking’ and press “done”.

3. Then press the “Scan event names” button (Figure 9B), and the previously large empty field named “Event List” (Figure 9C) should now be filled with different names for events that are present in the data set.

4. Open a spread sheet editor such as Microsoft Excel or OpenOffice Calc.

    a. Select and copy all entries in the “Event list” by clicking in the event list and pressing Ctrl+A followed by Ctrl+C, and paste the list into the spread sheet editor.

    b. Now we remove the events we are not interested in modifying from the spread sheet. That should be all rows except the top one and the ones that are named something with either ‘easy’, ‘medium’ or ‘difficult’.

    c. We also remove the entire column named ‘variable0’ since we are not interested in modifying those entries.

    d. Add a new column called ‘name’ next to the one that should already be present. For every row with an entry named something with “difficult” in the original column

named ‘name’, enter just ‘difficult’ in the new column.

e. Make the analogous entries for the rows named something with ‘easy’ and ‘medium’.

f. Select all active cells in the spreadsheet and copy them.

5. Move back to the **events_modification** window, clear the “Event list” field and then paste the entries from the spread sheet. It should now look like the window in **Fel! Hittar inte referenskälla.**.

A complete list of how the entry into this field should look can be seen in Appendix A, which can be cut and pasted into the event list field (instead of step 4 above).

6. Press the “Save as” button and save the setting as ‘pupil’.

7. Press “Use plugin” to add the plugin to the “Plugin workflow” in the TimeStudio main window.

Back in the main window (Figure 11)

8. Select all subjects and the **events_modification** plugin, then press “Do selected work” button to apply the modifications to the events.

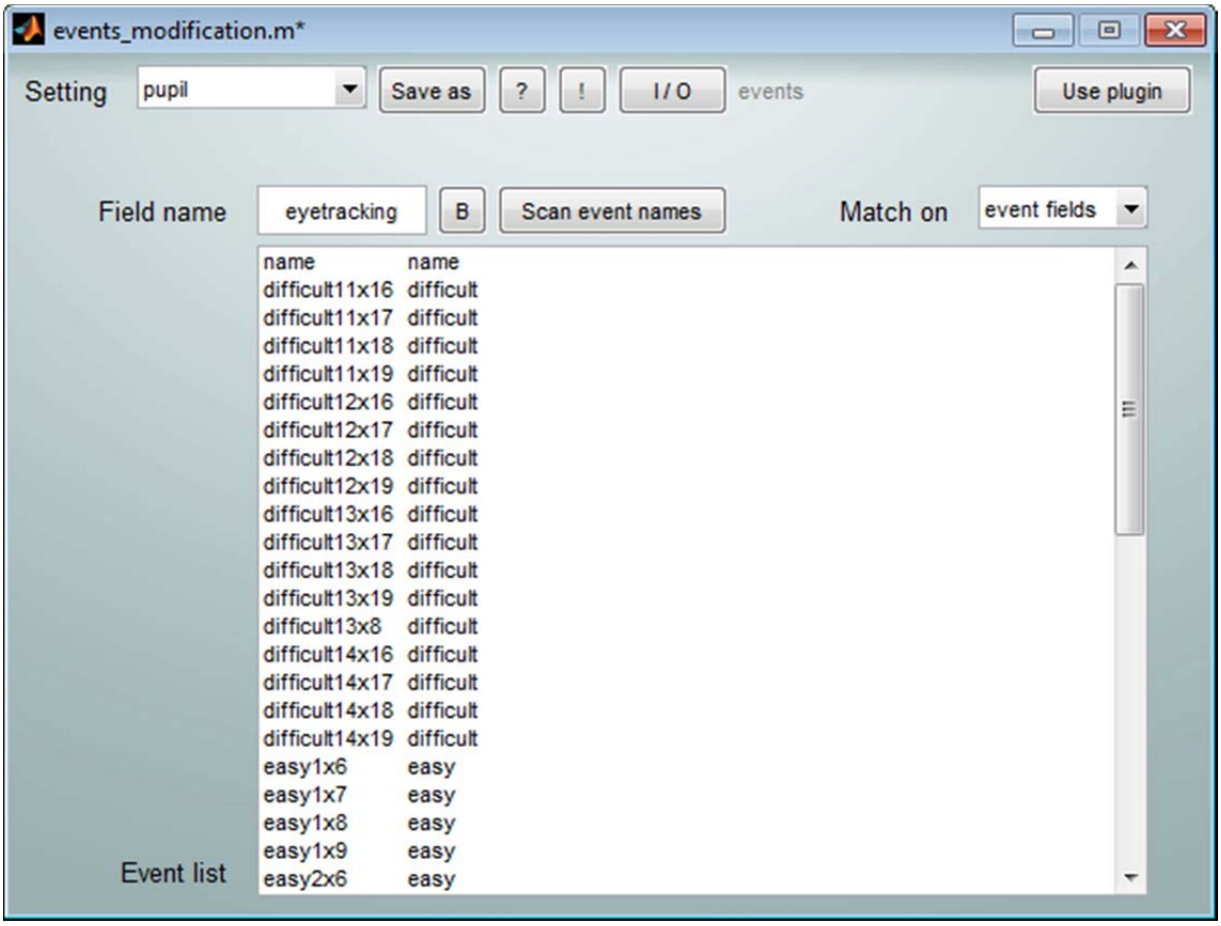

Figure 10. The setting window for the event_modification plugin with the settings we will use.

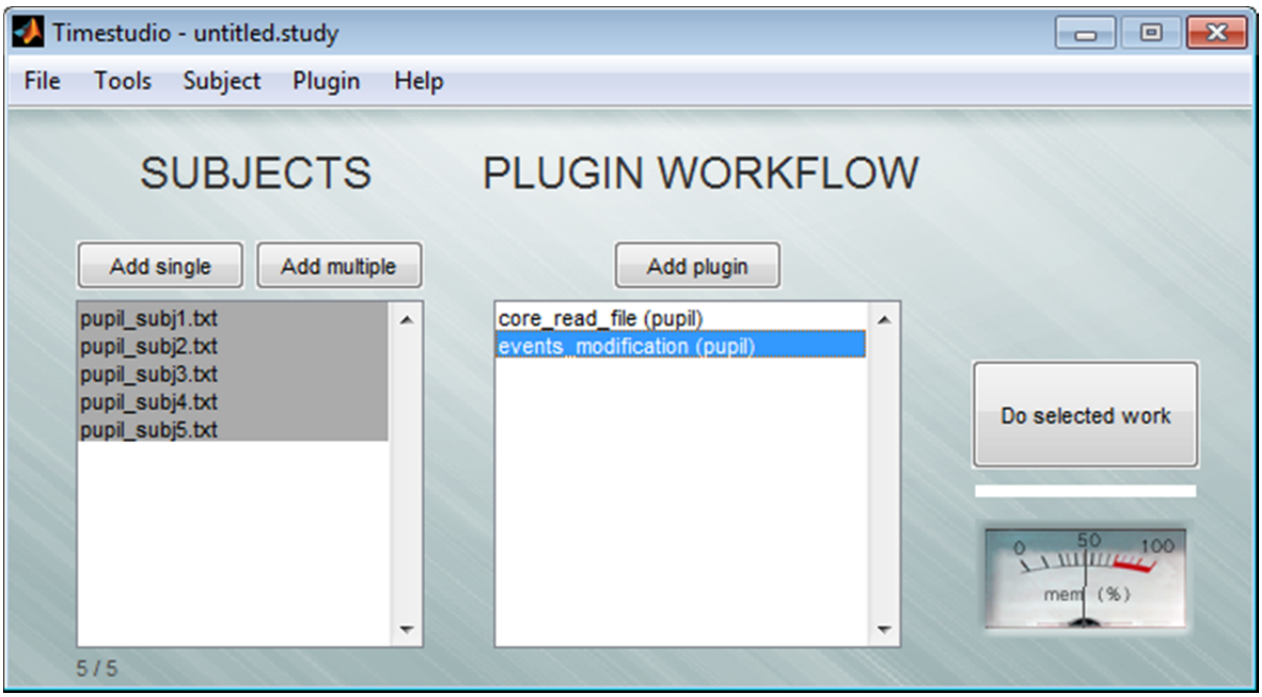

Figure 11. The TimeStudio main window after the plugin for event modification has been added.

1.5 Remove events with inadequate amount of data

The next step is to do some basic pre-processing of the pupil data. Some events contain a large amount of data with bad values. These are denoted by NaN (not a number). To remove events that contain a lot of bad data the plug-in **events_reject** can be used. In Figure 12 the plug-in is shown. This is the state after some values have been chosen.

1. As before, press “Add plugin” from the main menu to bring up the pop up for selecting plugins, press “events” and select **events_reject**.

2. Browse for data field at Figure 12A and select ‘eyetracking’.

3. Press the “B”-button at B in Figure 12. Select the event name ‘easy’. Now we have stated which events to study.

4. Enter the time range ‘0 8’ in position C in Figure 12, this will determine what interval around the event-time we will look at.

5. The next step is to select what is to be the X- & Y-data. Press the “B”-button and select ‘time’ for X series name (Figure 12D) and ‘pupil’ for Y series name (Figure 12E).

6. Now we can examine the data.
    a. By clicking on a rectangle the event will be plotted as an x-y plot in a separate window. See Figure 13 for example.
    b. The different events can be inspected by moving around using the arrows on the keyboard. By pressing space an event is rejected. This is shown by coloring that rectangle red. Pressing space again will un-reject it
    c. By pressing the ‘p’ button on the keyboard the x-y plot will plot the derivative of the ydata instead (actually it is the diff of y, see matlab for info on diff). Pressing ‘p’ again will switch back to ordinary y-data.

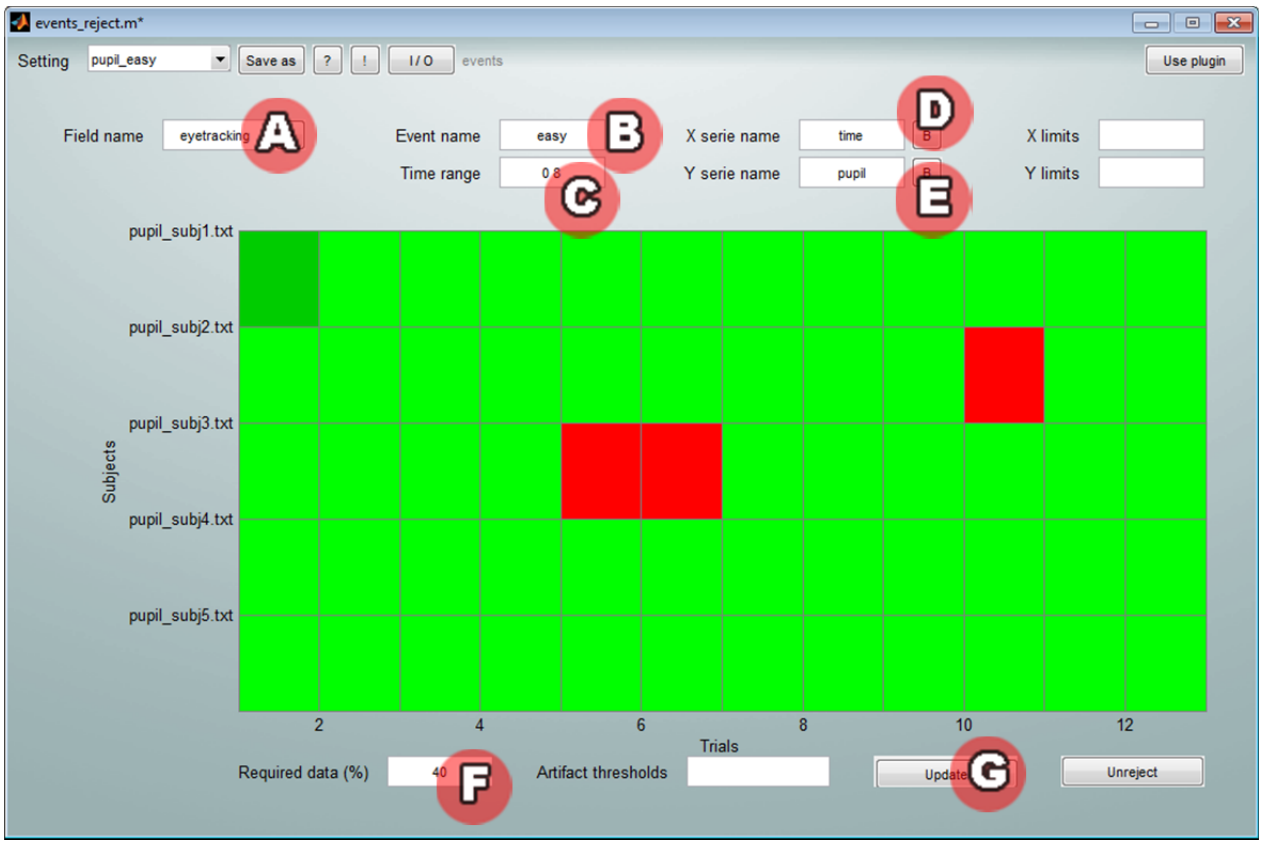

Figure 12. The event_reject window.

7. Instead of rejecting events manually we can do it in a more automatic way. Fill in the lowest acceptable percentage in the “Required data (%)”-box at position Figure 12F. A suitable number for this example is ‘40’. By pressing the “Update” button (Figure 12G) all events where the amount of bad data exceeds this percentage will be rejected (marked with red). When the plug-in is run in the workflow all the rejected events will be renamed with the string ‘_REJECTED’ appended to their name. This way they will not be part of the rest of the workflow.

8. Save the plug-in settings, like in the previous examples. A suitable name is ‘pupil_easy’. Then press “Use plugin”.

9. Now this procedure should be repeated for the other events, ‘medium’ and ‘difficult. Save the settings as ‘pupil_medium’ and ‘pupil_difficult’. The easiest way to do this is to

    a. Press “Add plugin” and select **events_reject.**
    b. Change the setting to ‘pupil_easy’.
    c. Change “Event name” to ‘medium’ by browsing with the “B”-button.
    d. Save as ‘pupil_medium’.
    e. Press “Use plugin”

    and then repeat for the ‘difficult’ events.

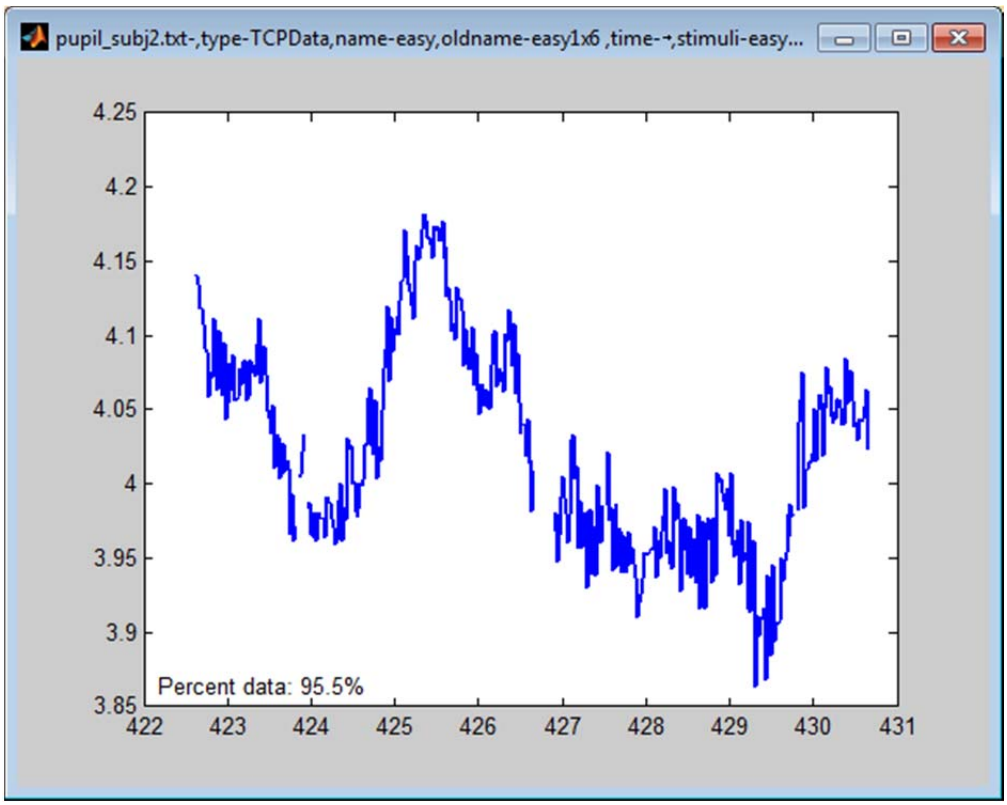

Figure 13. The inspection window from events_reject.

The TimeStudio main window should now look like Figure 14. Select the three **events_reject** and press “Do selected work”. A window that show how many events that were rejected will open. In our case, there will be three windows, one for each event (easy, medium and difficult).
For more ways to automatically reject data, see the plug-in documentation.

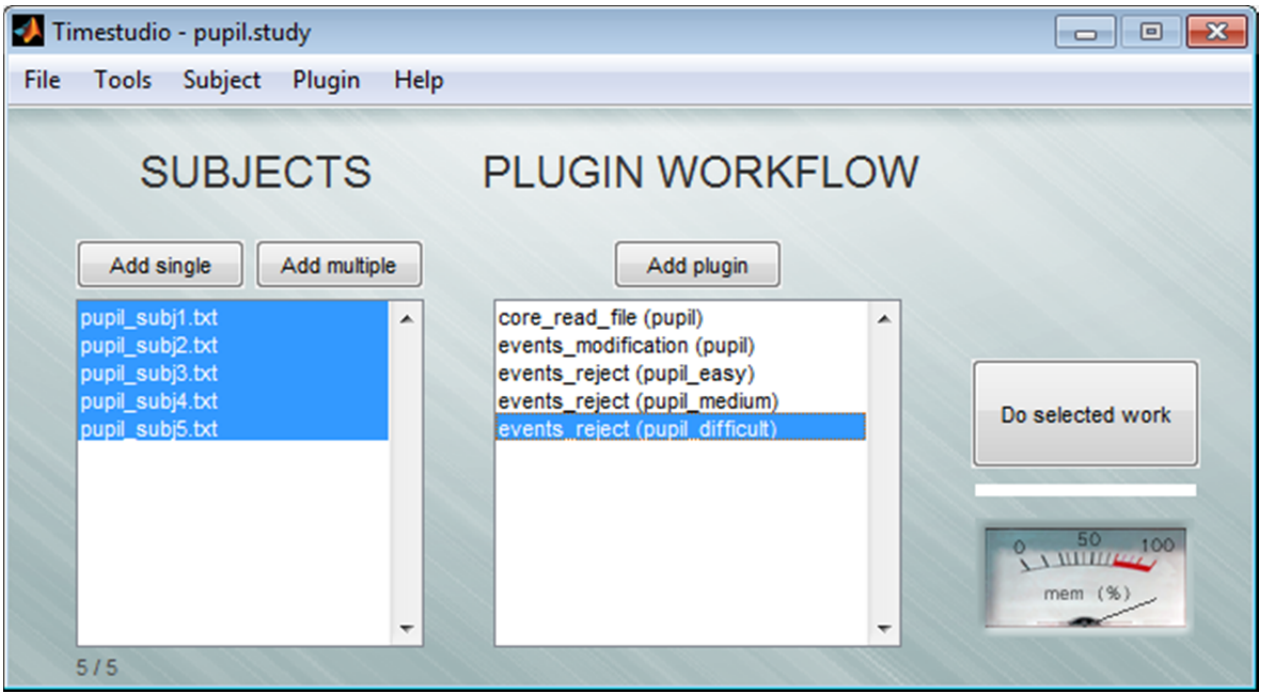

Figure 14. The TimeStudio main window after the events_reject plug-ins have been added.

1.6 Remove NaN-values

Although we removed the events with lots of missing data there are still evens that contain some bad data. These events will be handled by interpolation over the data points with bad (NaN – Not A Number) values. This is done by adding a plugin called **core_interpolate_gaps** to the study.

1. Press “Add plugin” and select the **core_interpolate_gaps** pluging. The plugin can be seen in Figure 15. Now we should enter the values shown there.

2. First browse for the data field (Figure 15A), and select ‘eyetracking’ in the pop-up window.

3. Browse for the data series (Figure 15B), and select ‘pupil’ in the pop-up window.

4. Enter ‘5’ into the “Max sample gap” field (Figure 15C) and make sure the boxes to the right (Figure 15D) are selected like they are in the figure.

This will interpolate over gaps with five or less consecutive bad data points. As always, more information about a plugin should be available from the plugin help.

5. Save as ‘pupil’ and press “Use plugin”.

6. In the TimeStudio main window, select all subjects and the **core_interpolate_gaps** plugin, and then press “Do selected work” (Figure 16).

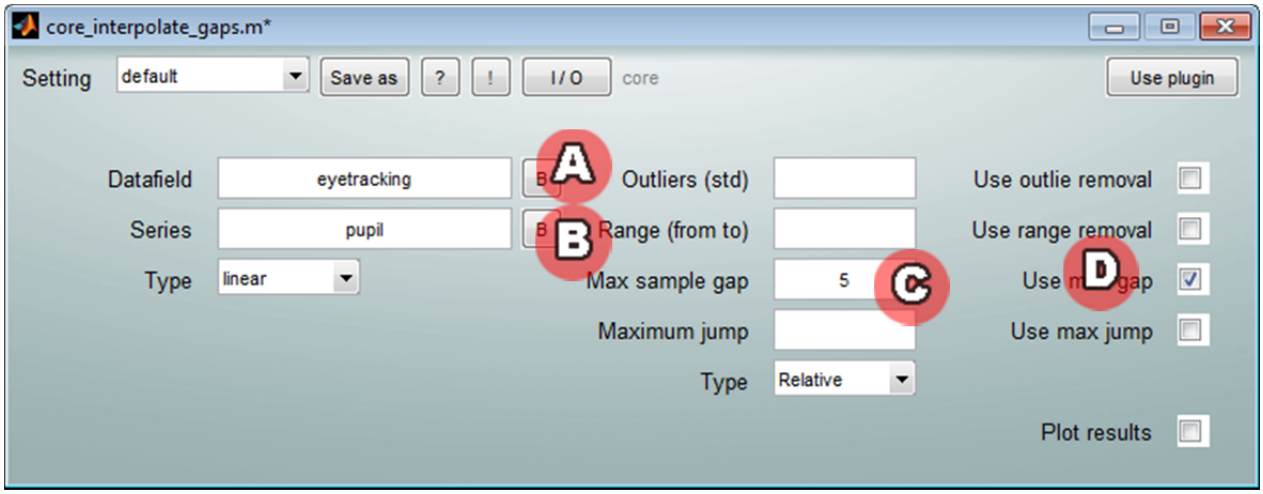

Figure 15. The setting window for the core_interpolate_gaps plugin.

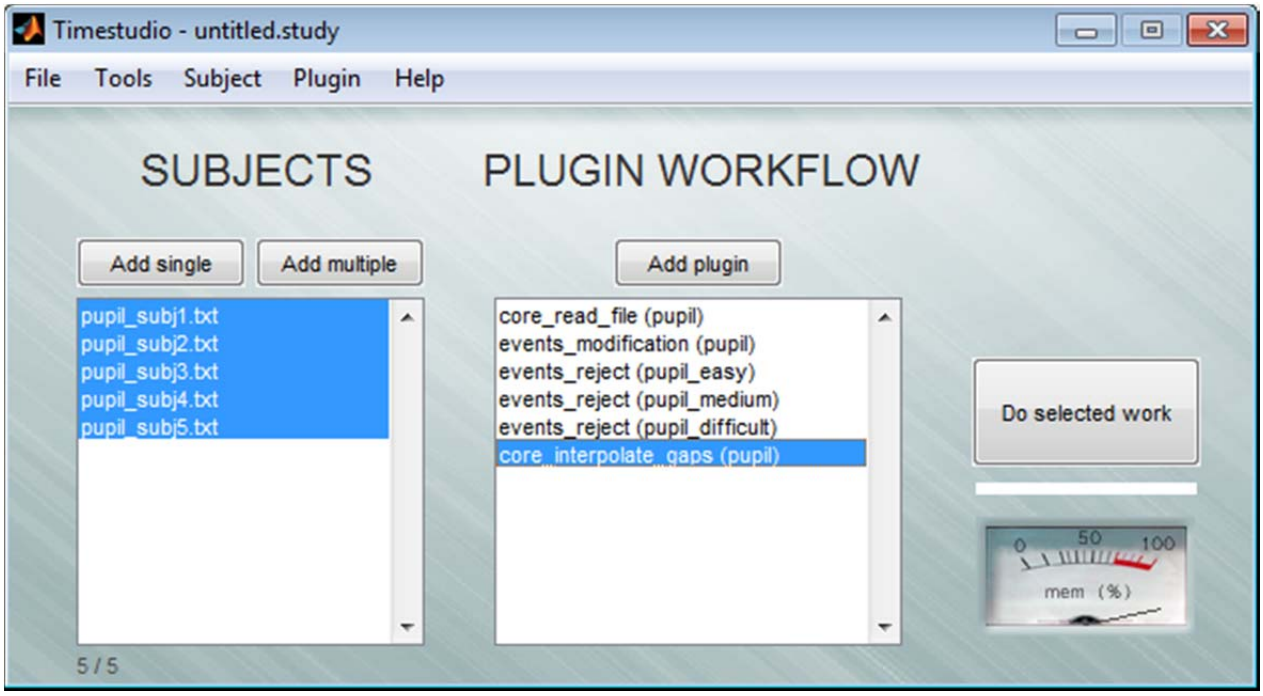

Figure 16. The TimeStudio main window after the plugin for interpolation has been added.

1.7 Applying an averaging filter

Now we will apply a moving average filter that reduce noise and smooth the data.

1. Press “Add plugin” and select the **core_filter_moving_average** plugin. The plugin can be seen in Figure 17. We should now enter the settings seen there.

2. Browse for the data field (Figure 17A), and select ‘eyetracking’ in the pop-up window.

3. Browse for the data series (Figure 17B), select ‘pupil’ in the pop-up window.

4. Enter a value in the field “Moving average” (Figure 17C) which sets the window length of the moving filter (in this manual we use a window length of 5 samples).

5. Save as ‘pupil’ and press “Use plugin”.

6. In the TimeStudio main window, select all subjects and the **core_filter_moving_average** plugin, then press “Do selected work” (Figure 18) to apply the filter to the pupil data for all subjects.

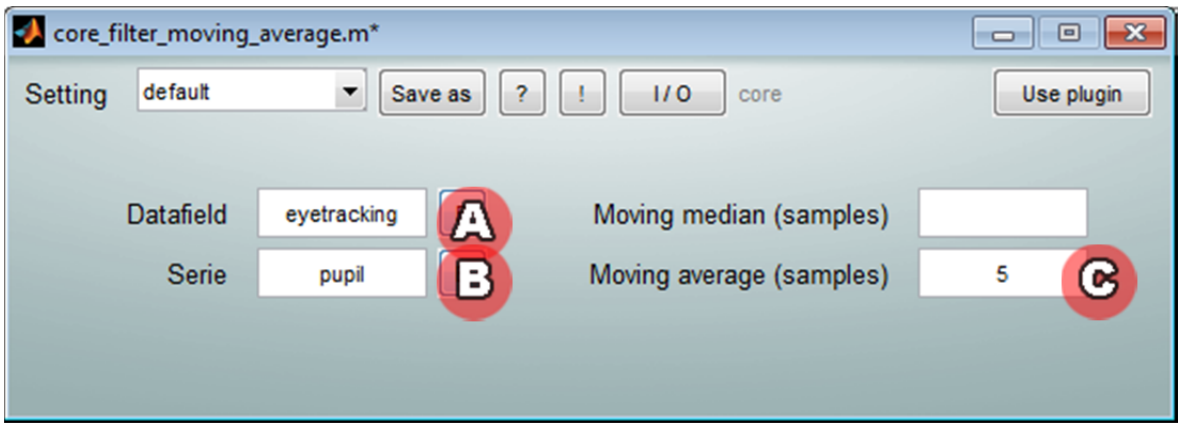

Figure 17. The setting window for the core_filter_moving_avarage plugin.

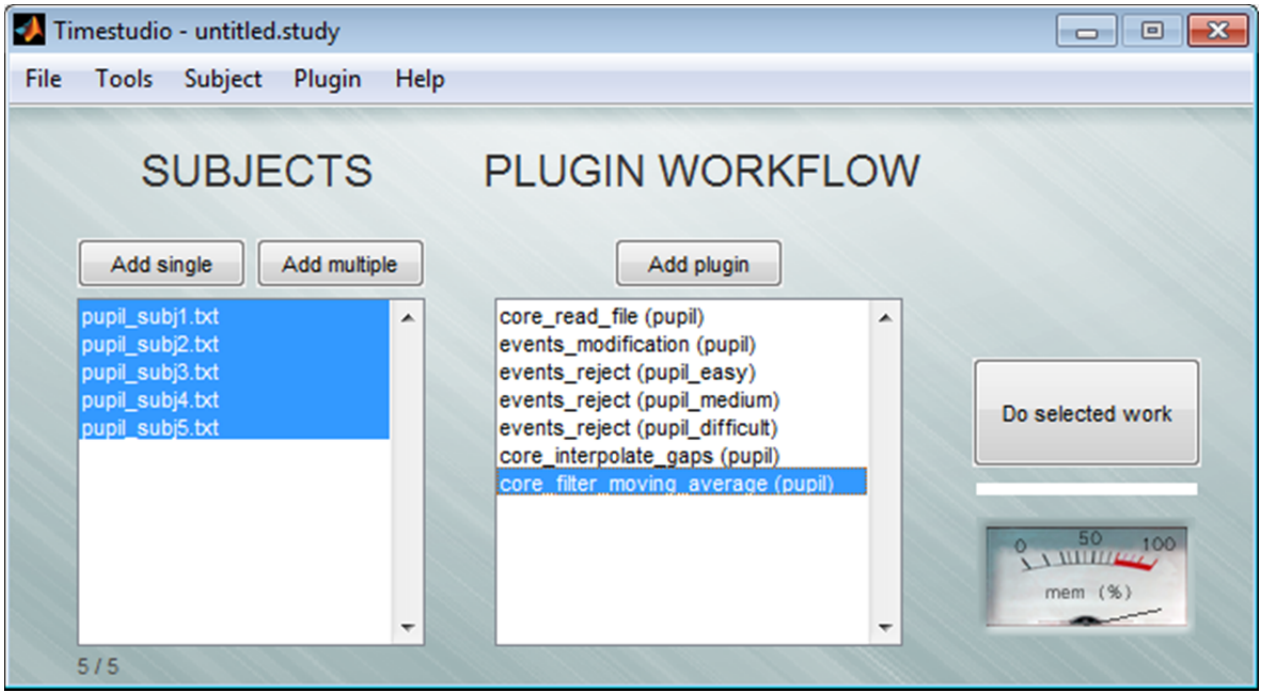

Figure 18. The TimeStudio main window after the plugin for filtering has been added.

1.8 Statistical extraction and visualization

The final step is the analysis of the data. This includes cutting out trials from the pupil time series (which are time locked to the events named easy, medium and difficult), baseline correcting each trial and calculating a mean pupil size at a particular time window in the trial. All this is done within a plugin called **core_event_related_data_extraction**.

1. Press “Add plugin” and select the **core_event_related_data_extraction** plugin. The plugin can be seen in Figure 19. We should now enter the settings seen there.

2. Browse for the data field (Figure 19A), select ‘eyetracking’ in the pop-up window.

3. Browse for the data series (Figure 19B), select ‘pupil’ in the pop-up window.

4. Browse for event names (Figure 19C), select the three entries ‘difficult’, ‘easy’ and ‘medium’.

5. Enter the values for “Time range” (Figure 19D), which is two values separated by a white space. The first number is how many seconds from the event start time each trial should start at. The second value is how many seconds from the event start time the trial should end at. Both positive and negative values are accepted, but the second value should always be larger than the first value to get trials with a positive time length. In this example we use the values ‘0 8’.

6. Enter the values for “Baseline” and “Analysis interval” (Figure 19E and Figure 19F) in the same manner as the time range: two values separated by a white space, meaning [from to] in relation to the start of the event. In this example we use the values ‘0 2’ for the baseline and ‘6 7’ as analysis interval.

7. Next, select the plot options (Figure 19G) and what to show (Figure 19H). In this manual we used ‘groups as figures’ and checked the boxes “Show mean” and “Show stderr”.

8. Save as ‘pupil’ and press “Use plugin”.

9. In the TimeStudio main window, select all subjects and the **core_event_related_data_extraction** plugin, then press “Do selected work” to apply the analysis on the subjects (Figure 20).

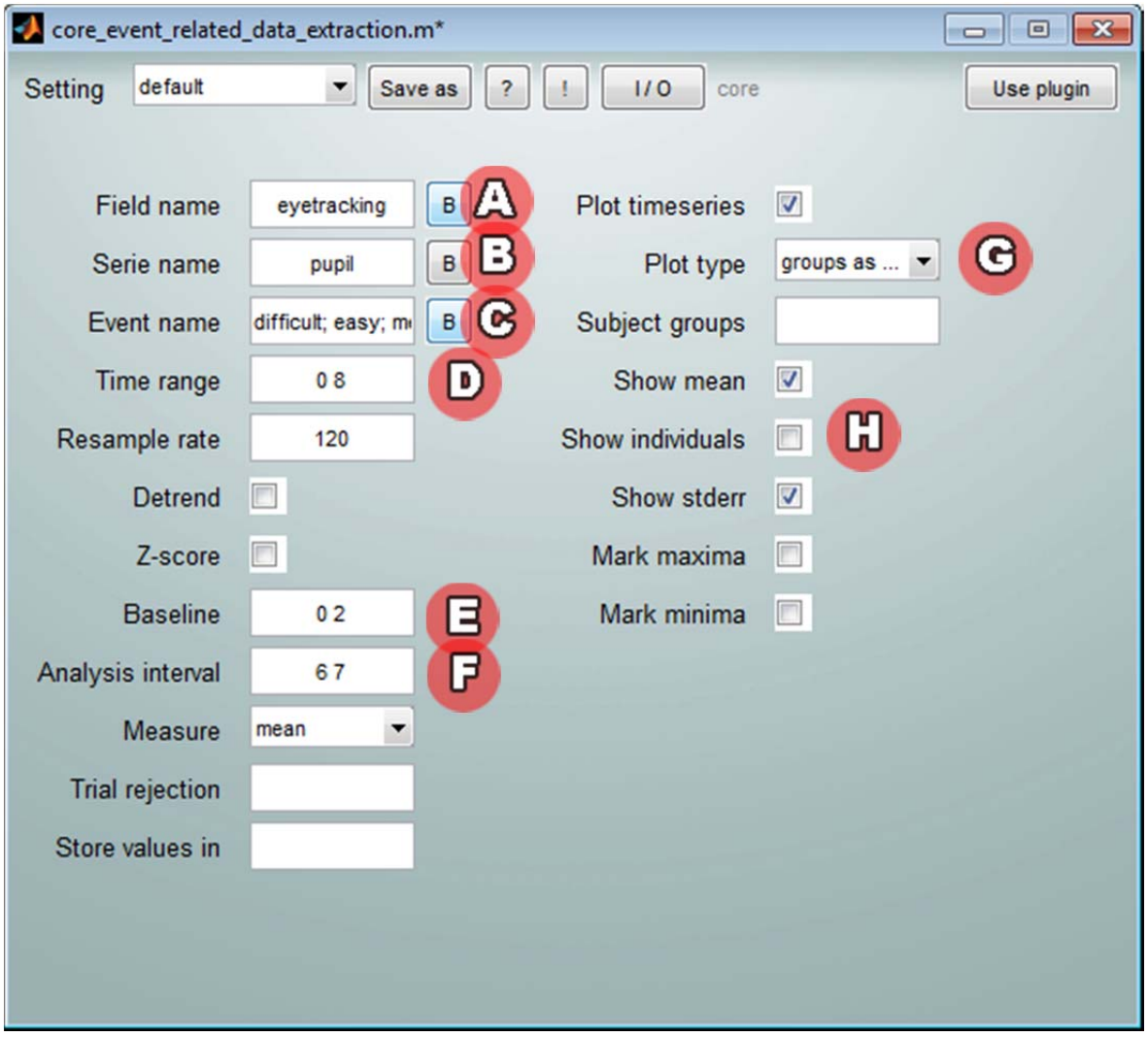

Figure 19. The setting window for the core_event_related_data_extraction plugin.

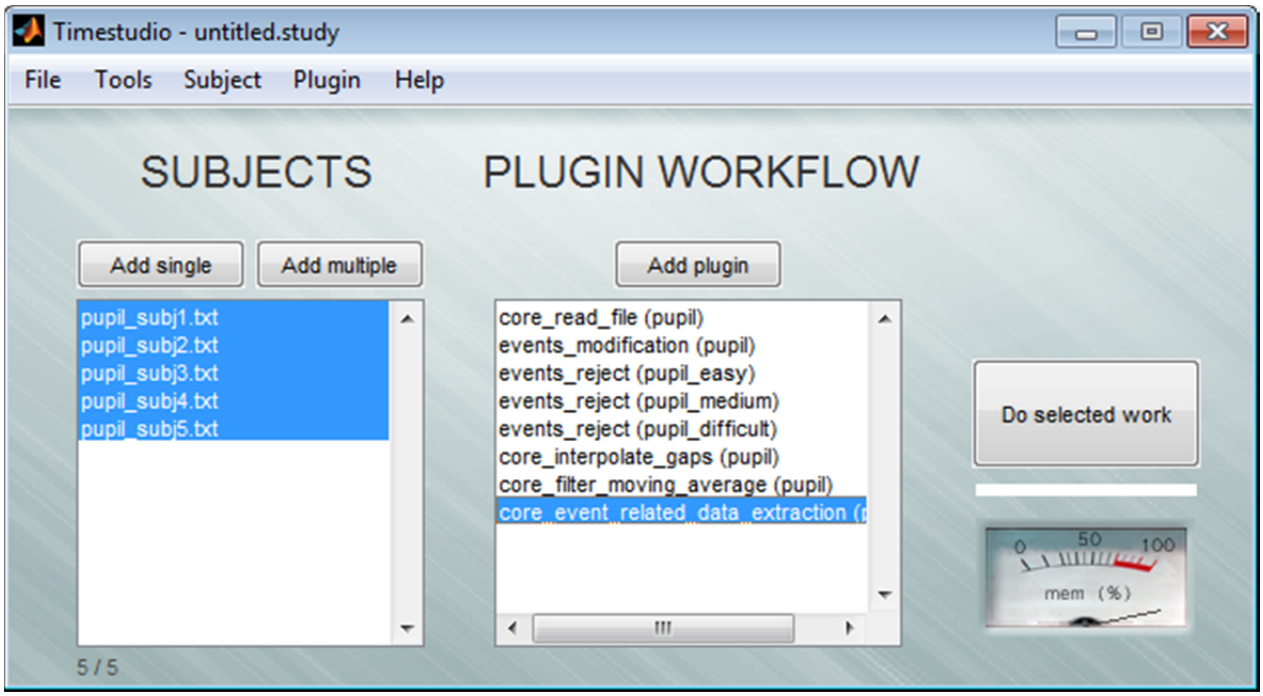

Figure 20. The TimeStudio main window after the plugin for analysis has been added.

1.9 View results

The final result can be seen in Figure 21and Figure 22. Note that the complete result is not visible in Figure 18 but the text extends well beyond the border of the window that can be seen in the figure. In this example the pupil dilates more (from the baseline) in the difficult condition than the medium condition. Pupil dilations are also higher in the medium than in the easy condition. Please compare this graph with prior findings by Ahern and Beatty (1979) in Figure 1.

Figure 21 shows the mean pupil size over time as a thicker line and the standard error as transparent patches. The results show that difficult trials dilate the pupil more than the easy ones. The shaded time interval is the analysis interval, and the mean pupil size in this time interval is calculated for each trial. The result of this calculation is shown in Figure 22.

Figure 22 is a text field with the mean pupil size in the analysis interval for all trials and all subjects. You can now copy all these values by clicking in the text field and pressing Ctrl+A followed by Ctrl+C. Paste the values into your favourite statistical analysis program (SPSS, Statistica, Excel etc.) to further analyse your data (remove outliers, do the statistical tests of your choice etc.).

Also worth mentioning is that during this demonstration all plugins has been applied one by one as they were added. It is also possible to first add all plugins (still in this particular order) to the study and then applying them all at once by selecting all subjects and all plugins and then pressing the “Do selected work” button.

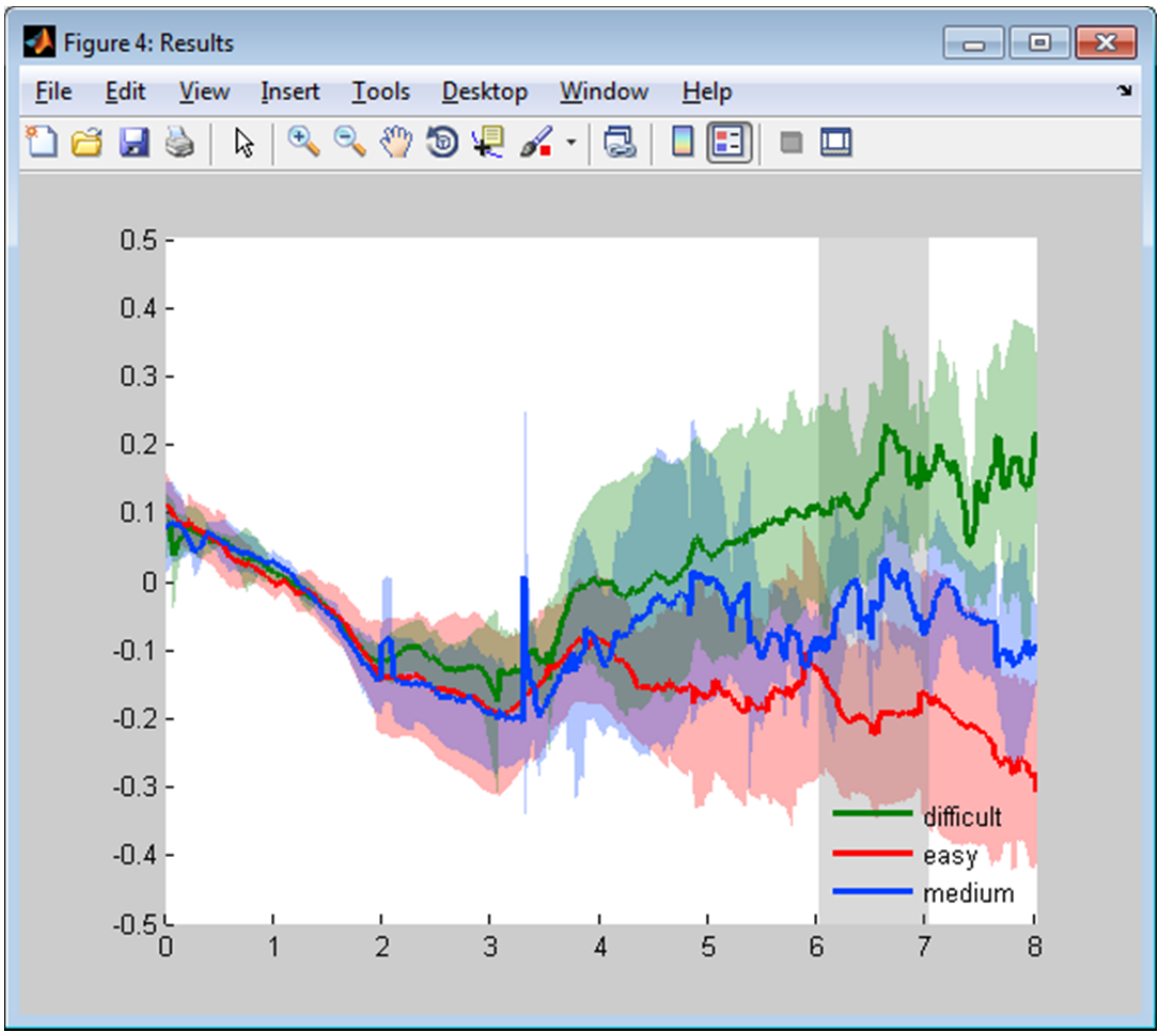

Figure 21 Plotted result.

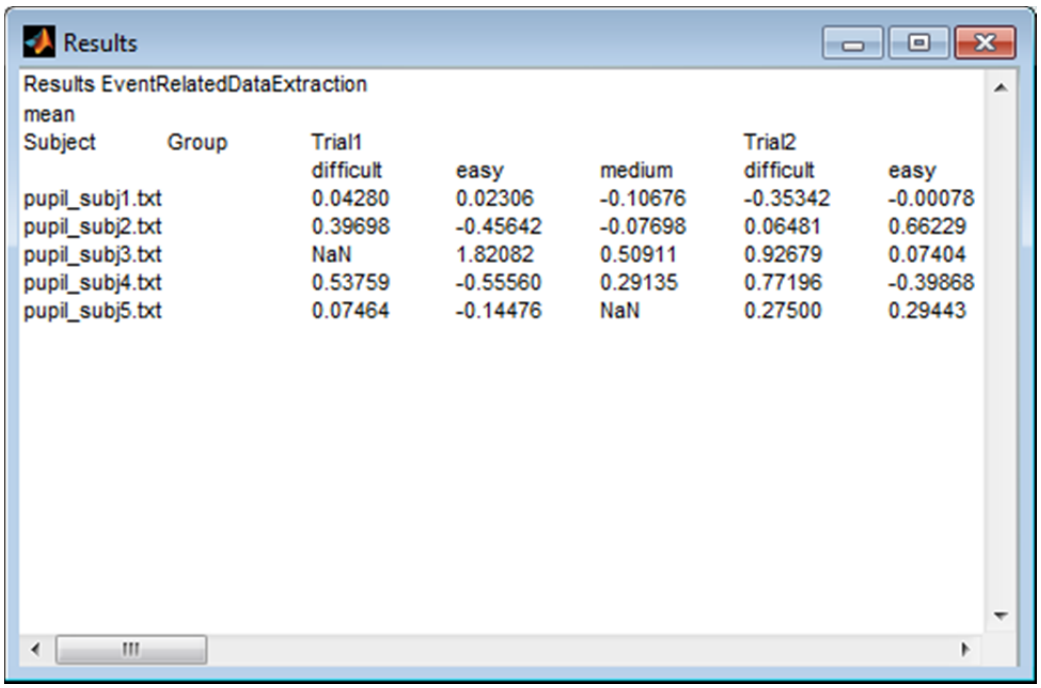

Figure 22 Results as text output.

4. References

Ahern, S., & Beatty, J. (1979). Pupillary responses during information processing vary with Scholastic Aptitude Test scores. *Science, 205*(4412), 1289-1292.

Laeng, B., Sirois, S., & Gredeback, G. (2012). Pupillometry: A Window to the Preconscious? *Perspectives on Psychological Science, 7*(1), 18-27.

5. Acknowledgements

The authors want to thank the following people for comments and suggestions for this manual and for testing the workflow: Gustaf Gredebäck, Terje Falck-Ytter, Joshua Jovrud , Tommie Forslund, Christine Fawsett and Ida Hensler. The manual was funded by grants ERC-StG- 312292-CACTUS and the Swedish Research Council in partnership with FAS, FORMAS and VINNOVA [Cross-disciplinary research programme concerning children's and young people's mental health; grant number 259 - 2012-24].

Appendix A – Entries for events_modification plugin used in 3.4

name	name
difficult11x16	difficult
difficult11x17	difficult
difficult11x18	difficult
difficult11x19	difficult
difficult12x16	difficult
difficult12x17	difficult
difficult12x18	difficult
difficult12x19	difficult
difficult13x16	difficult
difficult13x17	difficult
difficult13x18	difficult
difficult13x19	difficult
difficult13x8	difficult
difficult14x16	difficult
difficult14x17	difficult
difficult14x18	difficult
difficult14x19	difficult
easy1x6	easy
easy1x7	easy
easy1x8	easy
easy1x9	easy
easy2x6	easy
easy2x7	easy
easy2x8	easy
easy2x9	easy
easy3x6	easy
easy3x7	easy
easy3x8	easy
easy3x9	easy
easy4x6	easy
easy4x7	easy
easy4x8	easy
easy4x9	easy
easy8x3	easy
medium6x11	medium
medium6x12	medium
medium6x13	medium
medium6x14	medium
medium6x6	medium
medium7x11	medium
medium7x12	medium
medium7x13	medium
medium7x14	medium
medium8x11	medium
medium8x12	medium
medium8x13	medium
medium8x14	medium
medium9x11	medium
medium9x12	medium
medium9x13	medium
medium9x14	medium

Appendix B – Supported file formats for core_read_file

The **core_read_file** plugin can parse the following formats:

FileParser	FileFormat	Support
Tobii Text File	ClearView CombinedData text export	Full
Tobii Text File	TobiiStudio CombinedData text export	Full
Tobii Text File	TobiiStudio v.3 text export	Full
Eprime File	Eprime file format	Partial