**EXCEL MACROS:**

**DATA PREPARATION & ANALYSIS FOR SYNCHRONIZED TIME SERIES**

Adolescent Dynamics Lab – Macro Manual

August 2015

These macros have been created and refined in the Adolescent Dynamics Lab at Queen’s University in Kingston, Ontario from 2006 to the present. While most of the early macros were written by lab director Dr. Tom Hollenstein, there have been many students who have added original macros and enhanced older ones. Most recently, massive credit goes to Kate Jackson and Samantha Goldsmith who prepared these macros for sharing in 2015. However, credit and thanks also goes to: Jordan Theriault, Ellen (Big Mac) O’Donoghue, Effie Pereira, Jess Lougheed, Dianna Lanteigne. Where appropriate, these authors are identified in the descriptions below.

Any questions or comments please email: **macros@statespacegrids.org**

Table of Contents

[1) INTRODUCTION 4](#_Toc428451421)

[2) ALTERING FILE NAME OR FILE TYPE 5](#_Toc428451422)

[2.1) Adding Filename Ending 5](#_Toc428451423)

[2.2) Removing Filename Beginning or Ending 6](#_Toc428451424)

[2.3) Resave Excel as New Format 6](#_Toc428451425)

[3) TIME SERIES PREPARATION 7](#_Toc428451426)

[3.1) Adding Task Variables 7](#_Toc428451427)

[3.2) Customized Merging 9](#_Toc428451428)

[3.3) Logged Variable Creation 11](#_Toc428451429)

[3.4) Missing Value Insertion, Removal, Replacement 12](#_Toc428451430)

[3.5) Onset Correction 13](#_Toc428451431)

[3.6) Renaming Columns 13](#_Toc428451432)

[3.7) Removing Excess Time Series Data 14](#_Toc428451433)

[3.8) Restricting Range of Time Series Values 16](#_Toc428451434)

[3.9) Rounding Data 16](#_Toc428451435)

[3.10) Creating Binned Data 17](#_Toc428451436)

[3.11) Categorization 18](#_Toc428451437)

[4) APPENDING TIME SERIES DATA 22](#_Toc428451438)

[4.1) Remove Empty Row 22](#_Toc428451439)

[4.2) Adding Participant ID Column 22](#_Toc428451440)

[4.3) Adding Columns of Variables with Constant Values 23](#_Toc428451441)

[4.4) Append Full Time Series 25](#_Toc428451442)

[4.5) Stacking Variables and Dummy Coding (for SPSS Mixed Models and Multilevel Modeling) 28](#_Toc428451443)

[5) CREATING NEW VARIABLES 32](#_Toc428451444)

[5.1) Baseline Deviations 32](#_Toc428451445)

[5.2) Difference Scores 34](#_Toc428451446)

[5.3) Logged Variable Creation 34](#_Toc428451447)

[5.4) Lagged Variable Creation 35](#_Toc428451448)

[6) GRAPHING 36](#_Toc428451449)

[6.1) Single Channel with Dynamic Axes and Task Coding 36](#_Toc428451450)

[6.2) Single Channel with Specified Axes Values 39](#_Toc428451451)

[6.3) Two Channels with Specified Axes Values 40](#_Toc428451452)

[7) CORRELATIONS 41](#_Toc428451453)

[7.1) Within-Subject Zero-Order Correlations 41](#_Toc428451454)

[7.2) Customizable Lagged Correlations 42](#_Toc428451455)

[7.3) Windowed Correlation Time Series 43](#_Toc428451456)

[8) Noldus Observer Event Logs to TRJ Conversion 44](#_Toc428451457)

# 1) INTRODUCTION

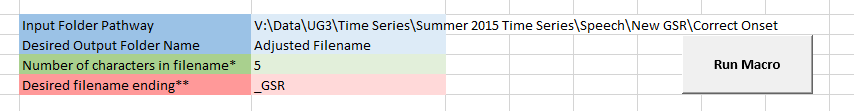
These macros were designed to conduct a variety of manipulations on synchronized time series data. They were developed primarily for working with continuous data (e.g., heart rate) but can also be used with categorical data (e.g., code categories). These macros are also useful for combining Gridware data files (\*.trj) with other synchronized data of similar format (files wherein the first column is an “Onset” column delineating the time or integer counts of events). Most often, the first row of data has an Onset of zero, though this does not have to be the case. Subsequent columns (columns 2 and beyond) contain state variables or values of a time series. For GridWare, these values must be integers or text; however, as previously mentioned, these macros are designed to also work with continuous values. Therefore using these macros, both continuous and categorical variables can be included and formatted for analyses outside of GridWare (e.g., multilevel modeling).

In most cases, these macros assume you are starting with a folder containing two or more text files, with each text file corresponding to one individual participant. All assumptions for input files and their contents, format, etc. will be indicated where appropriate for each macro.

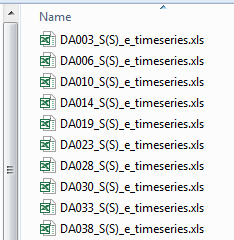
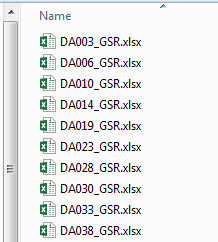
# 2) ALTERING FILE NAME OR FILE TYPE

## 2.1) Adding Filename Ending

Adds text to the end of all filenames in a folder based on user input and saves the resulting new files with revised names in a new folder. For example, user can add “\_HR” to all heart rate time series files. This macro was designed to solve the problem encountered when trying to merge two files of the same name (i.e. if both the HR and GSR files are called “AB123”). In the example below, since we only entered “5” for the number of characters in the filename, only the first five characters were kept. The remaining text was deleted and replaced with our desired ending, “\_GSR”.

**Input:**  


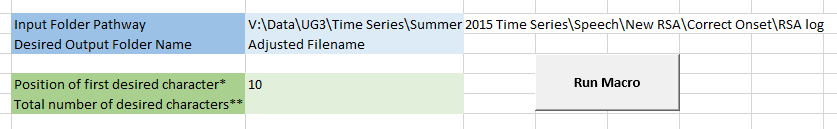
**Original Files:**  **Resulting Files:**

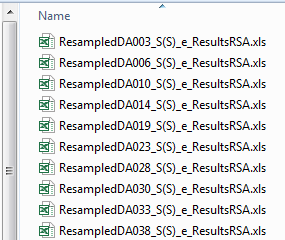
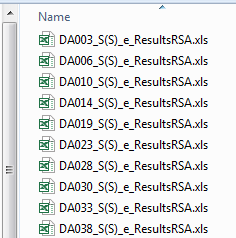
 

## 

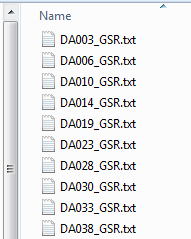
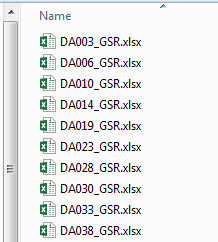
## 2.2) Removing Filename Beginning or Ending

Extracts the participant ID (or any desired set of characters) from within a filename and saves newly renamed files in a new “desired output” folder. User can remove characters from a filename beginning, ending, or both simultaneously. This macro was designed to solve the problem encountered when trying to merge two files that do not both have the participant ID at the beginning of the filename (i.e. trying to merge AB123\_HR and ResampledAB123\_RSA). If the “Total number of desired characters” cell is left blank, all text to the **right** of the specified starting point will be kept. If you wish to only include a few characters (i.e., just the participant ID), you can enter the appropriate value (i.e., 5) in the input sheet.

**Input:**

**Original Files: Resulting Files:** 

## 2.3) Resave Excel as New Format

Opens each Excel file and either resaves it as a tab-delimited text file (.txt) or exports it as a PDF file (.pdf), depending on the button selected. In both cases, the original files are **not** permanently altered and the new files are saved under a new folder.  
  
**Original Files: Resulting Files:**

# 3) TIME SERIES PREPARATION

## 3.1) Adding Task Variables

Adds a column containing numeric values that specify which part of an experiment is being completed at each second of a time series. This numeric task column is added to the first free column on the right of each text file within the input folder. For example, if a participant is listening to instructions for the first 15 seconds of a time series, this macro will create a new “Task” column and assign 0’s to the first 15 rows. The values will then switch to 1’s and then 2’s as the task changes.

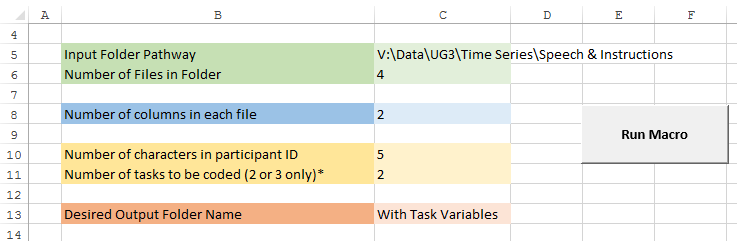
The number of rows for each task is calculated using the user input in Sheet2 of the macro. The user inputs time values (in minutes and seconds) for each task. The workbook will automatically convert these values to seconds and then use these values to determine how many seconds are included in each task. The macro then uses the calculated values (in columns O, P, and Q) to insert numerical values in the “Task” column.

You may also enter values into columns O, P, and Q manually, however the equations executed by these cells will be erased. It is recommended that when entering values in manually, you copy and paste the existing columns O, P, and Q to any region beyond Column Q, allowing you to re-insert them afterwards for future use with the time calculations.

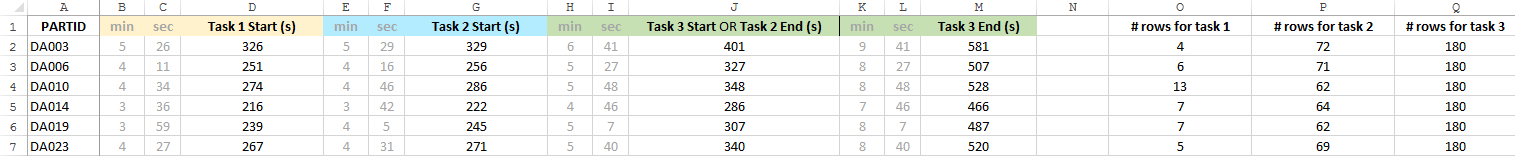
**Note:** Do NOT rename the macro workbook or the names of Sheet1 and Sheet2 as the code itself would also have to be altered to recognize the new names.

**Assumptions:** This macro assumes a minimum of two tasks and a maximum of three tasks. If using only two tasks, Column K, L, and M in Sheet2 should correspond to the **end** of Task 2 (rather than the **start** of Task 3). If using more than three tasks, this macro will require Visual Basic editing.

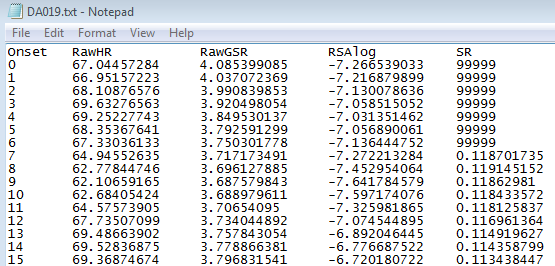
**Input:**



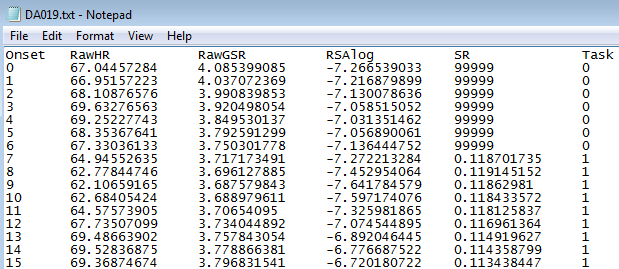
**Input (Sheet2):**

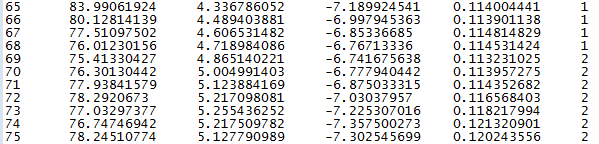


**Example of One Original File**



**Resulting File:**





Note that in this example (DA019), Sheet2 specified a total of 7 rows for Task 1, and the “Task” column includes 0’s for the first 7 rows of the time series. Similarly, there should be 62 rows for Task 2, and the “Task” column inserted 1’s for a total of 62 rows (ranging from an onset time of 7s to an onset time of 68s).

## 3.2) Customized Merging

Combines the data from multiple different time series files by copying from one file and pasting into another. For example, can copy the GSR data from participant AB123 and paste it as a new column alongside the HR time series for participant AB123. The files you are *copying from* should be within Folder 1, and the files you are *pasting to* should be within Folder 2. The resulting merged files are saved to a new folder within Input Folder 2. They are also renamed using only the participant ID. For example, when merging AB123\_HR and AB123\_GSR, the resulting file is saved as “AB123”.

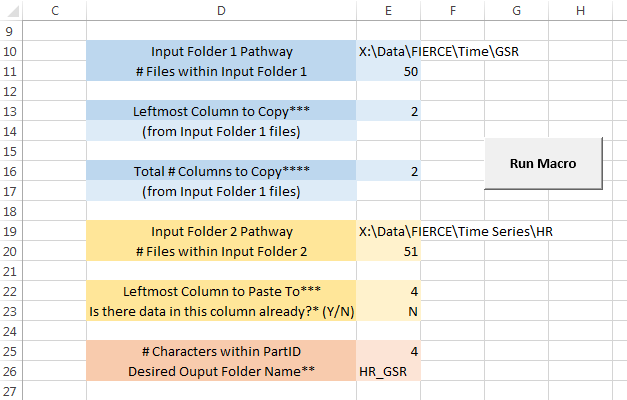
The macro ensures that the files being combined have the same participant ID. Therefore it is essential that the name of each file **begin with** the participant ID. If there are other characters at the beginning of the file names, use the “Removing Filename Beginning” macro ([Section 2.2](#_2.2)_Removing_Filename_1)) first. It is also essential that the files being merged do **not** have identical names. For example, even if they are saved within different folders, they cannot both be called “AB123.txt”. If this is the case, use the “Adding Filename Ending” macro ([Section 2.1](#_2.1)_Adding_Filename)) first to add an identifying ending (i.e. add “\_HR” to all of the HR files).

For participants who have a file in Folder 1 but not Folder 2 or vice versa (i.e., the participant has a GSR time series but no HR time series), the macro is designed to replace the entire missing column with missing values. In order for this section of the code to function properly, the number of files within each input folder must be specified correctly in the input sheet.

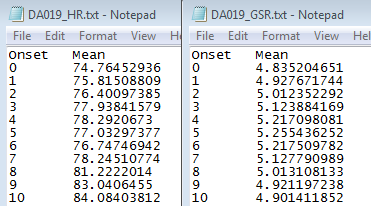
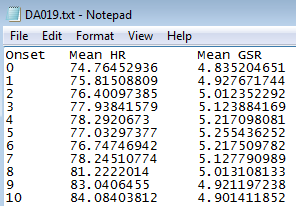
This merging process can be repeated for any number of files. New columns will be added to the first available column on the right in each existing file. When adding new columns to an existing merged file (i.e., adding an RSA column to the merged files containing HR and GSR), the merged files should always be in Folder 2. Note that the newly merged files are always saved using the Participant ID as the filename (e.g., AB123.txt), therefore any new files you wish to merge must have some additional text at the end (e.g., AB123\_RSA).

**Assumptions:**

* File names must begin with the participant ID (if this is not the case refer to [Section 2.2](#_2.2)_Removing_Filename_1)).
* File names must **not** be identical (if they *are* identical, refer to [Section 2.1](#_2.1)_Adding_Filename)).
* Files being merged must have the same onset. The onset value does **not** have to be zero; however, if the files in Folder 1 begin at 1 second, the files in Folder 2 must also begin at 1 second. Similarly if the files in Folder 1 have been onset-corrected ([Section 3.5](#_3.5)_Onset_Correction)) the files in Folder 2 must also be onset-corrected prior to merging.
* If files being merged are of **different** lengths (i.e., merging a 180-second GSR time series and a 165-second RSA time series), always ensure that the longer files are within Folder 2 and the shorter files are within Folder 1. Again, the onset value must be the same for both. It is also recommended that you merge all longer columns first and then begin adding the shorter columns. The bottom of the newly merged files will have blank spaces for time points where there is data in one column but not the other. For example, when merging a 180-second GSR time series and a 165-second RSA time series, the last 15 rows of the merged file will contain a value in the GSR column and a blank cell in the RSA column. The “Insert Missing Values” macro ([Section 3.4](#_3.4)_Missing_Value_1)) can be run on all merged files to replace these blank cells with missing values.

**Input:**

**Example of Two Separate Original Files: Resulting Merged File:**

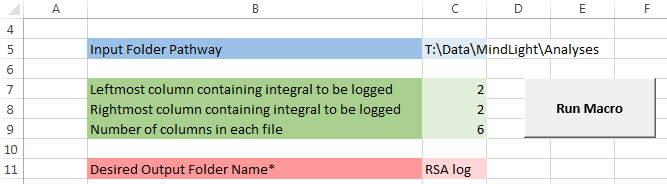
 

Note that these are originally two separate time series, one HR and one GSR. The resulting file includes just one onset column, with the HR and GSR data pasted side-by-side in a single text file. Also note that the names of the two input files both begin with the participant ID (DA019) and do **not** have identical names.

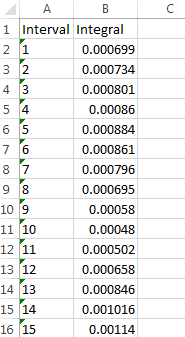
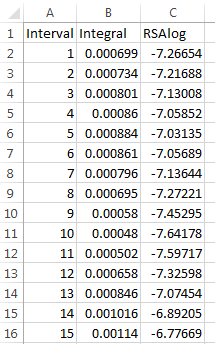
## 3.3) Logged Variable Creation

Takes the natural log of all values within a specified column or range of columns and pastes these logged values into the first available column. This macro was designed to get correct RSA values from the raw integral time series exported from our psychophysiological analysis software (AcqKnowledge). If you wish to take the log of just one column, enter the same value for the leftmost and rightmost column to be logged (Cells C7 and C8).

**Input:**



**Example of One Original File: Resulting File:**

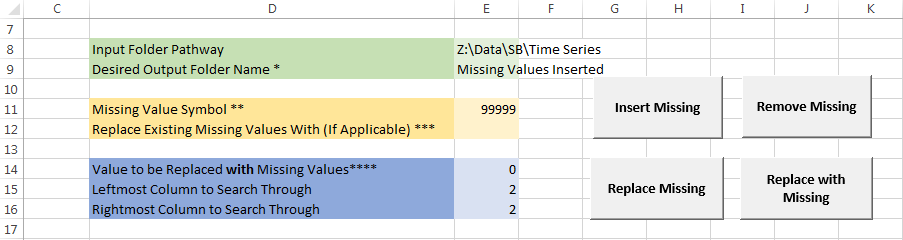
## 

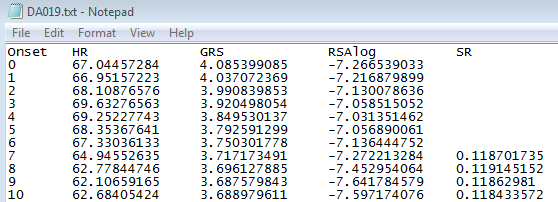
## 3.4) Missing Value Insertion, Removal, Replacement

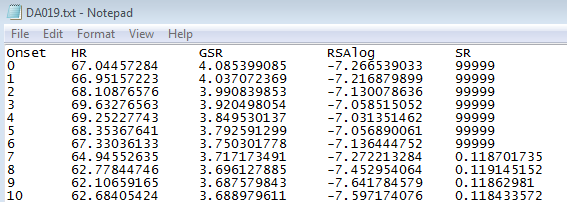
This workbook contains four slightly different macros that work with missing values. The symbol for missing values (e.g. 99999) must be specified by the user.

**Insert Missing:** Replaces all blank cells with missing values.  
**Remove Missing:** Replaces all missing values with blank cells.  
**Replace Missing:** Changes all missing values to a user-specified value.  
**Replace with Missing:** Replaces all occurrences of a user-specified value with missing values. This macro will only search the specified columns (i.e. if you want to replace all 0’s with missing values, can ensure the 0 in the onset column is excluded).

When inserting or removing missing values, only cells E8, E9 and E11 must be filled out (all others can be left blank). When replacing missing values, only cells E8, E9, E11 and E12 must be filled out. When replacing some value with missing values, cells E8, E9, E11, E14, E15 and E16 must be filled out.

**Input:**  


**Example of One Original File:**

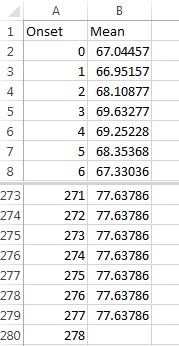
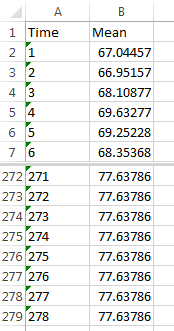
**Resulting File (Using “Insert Missing”):**

## 3.5) Onset Correction

Inserts a time value of 0 (representing the *onset* of the physiological response) and shifts all physiological data up by one cell. The final row in the time series will therefore be blank. The macro loops through all files within the input folder and corrects the onset of each one.

This macro is generally the first step when editing time series files, and **must** be done before running certain other macros (e.g., removing excess data). If onset must be corrected for a macro to run properly, this will be clearly indicated in the description.

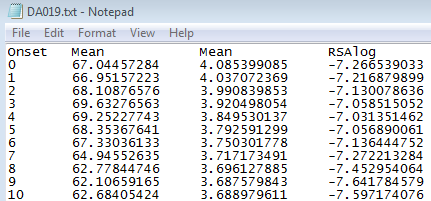
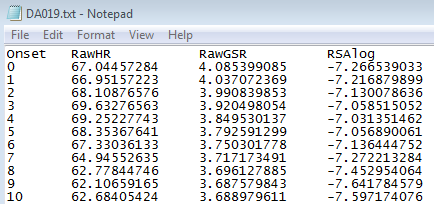
**Assumptions:** This macro requires all files within the input folder to begin with the same onset value prior to running the macro. It also requires that all files have fixed intervals. That is, that each row in the file corresponds to the same time interval (e.g., one second).

**Example of One Original File: Resulting File:**

## 3.6) Renaming Columns

Changes the names of all columns within a time series. Files in the input folder may contain up to 15 columns in total. The macro changes the names of all columns, so if you wish to keep an existing header you must re-enter the title. Blank input cells will result in no column header in the new file.

**IMPORTANT:** This macro saves over the original file.

**Example of One Original File: Resulting File:** 

## 3.7) Removing Excess Time Series Data

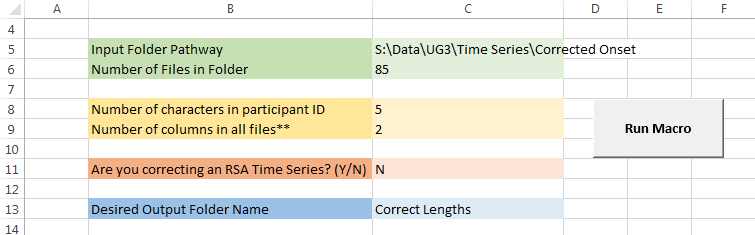
Shortens time series files to match the desired lengths (as specified by the user). This macro is used when the total length of a time series file is different for each participant (for example, when the instructions before a task are included in the time series). The Acknowledge scripts used to obtain the time series files are run for all participants using the longest total time, therefore the excess data must be deleted afterwards.

The individual file lengths are calculated using data from Sheet2. The user inputs time values (in minutes and seconds) for different sections of the experiment. The workbook then converts these values to seconds and uses them to determine the total length for each participant. In the example below, the user inputs the Instruction (Task 1) Start time and the Speech (Task 2) Start time, allowing the workbook to calculate the total length of the instructions, which will be different for each participant. In Column L, this instruction length value is added to the total length of the speech to create a total time. This is the final value used in the macro to remove the necessary number of rows.

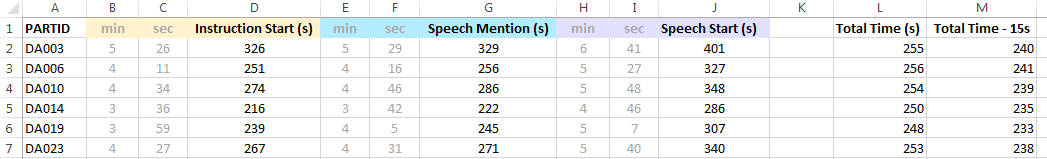
Note that when correcting the RSA time series, the total length is shortened by 15 seconds because of the limitations of the 16-second window method used to obtain the RSA time series. Therefore when correcting an RSA time series (which the user specifies in the input sheet), the macro uses Column M instead of Column L. In both cases, the macro matches the Participant ID from each filename to the Participant ID in Column A of Sheet2 and uses the corresponding length.

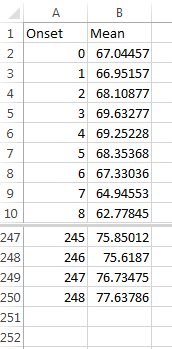
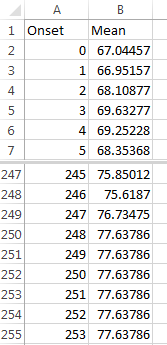
You may also enter values into columns L and M manually, however the equations executed by these cells will be erased. It is recommended that when entering values manually, you copy and paste the existing columns L and M to any columns beyond Column M so that you can re-insert them afterwards for future use.

**Note:** Do NOT rename the macro workbook or the names of Sheet1 and Sheet2 as the code itself would also have to be altered to recognize the new names. **Input files must already have corrected onset.** That is, the first row of data should correspond to an onset time of 0 seconds. If this is not the case, run the “Onset Correction” macro ([Section 3.5](#_3.5)_Onset_Correction)) first.

**Input:**

Note that since we input “N” in Cell C11, the total time value will be taken from Column L of Sheet2.

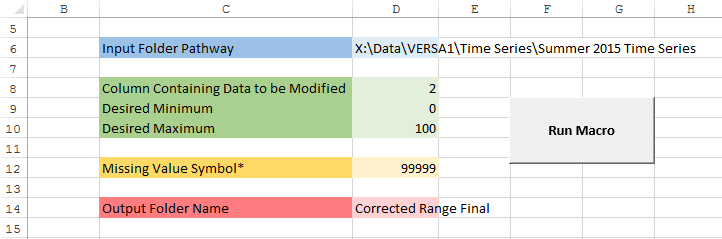
**Input (Sheet2):**

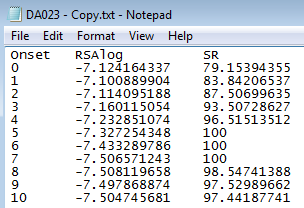
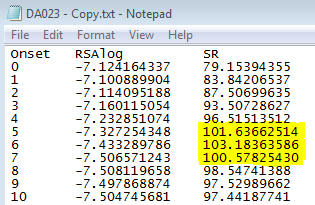
**Example of One Original File: Resulting File:**

These files correspond to participant DA019. Note that in the original file, the values start repeating after 248s. This corresponds with the value in Column L of the input sheet, where participant D019 should have a total length of 248s. The macro uses this value and removes all data below time 248.

## 3.8) Restricting Range of Time Series Values

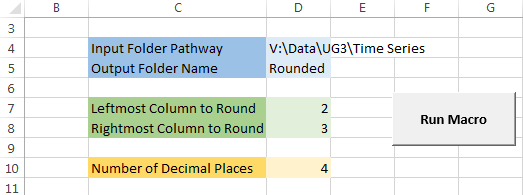
Restricts the range of values in a time series by altering any numbers below a specified minimum and above a specified maximum. All values below the specified minimum will be replaced with that desired minimum value. All values above the specified maximum will be replaced with that desired maximum value. All other values – including missing values – will be unaffected. For example, if working with self-report files where values should range from 0 to 100, all negative values will be increased to zero and all values greater than 100 will be decreased to 100.

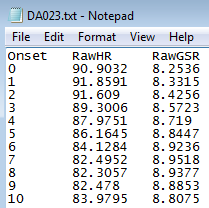
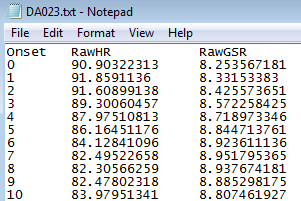
**Input:**

**Example of One Original File: Resulting File:**

## 3.9) Rounding Data

Rounds all values within a specified column or range of columns to a specified number of decimal places.

**Input:**

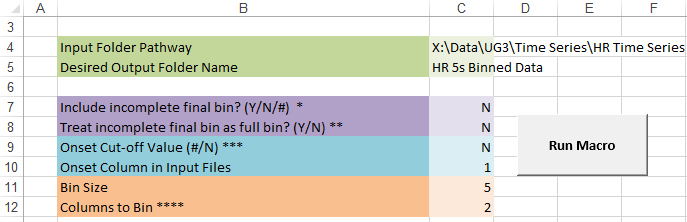
**Example of One Original File: Resulting File:**  


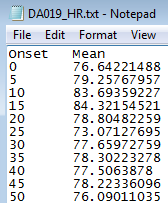
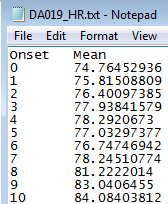
## 3.10) Creating Binned Data

Creates a time series of binned values by taking the weighted average across each bin of a user-specified size. For example, if user enters a bin size of 30 seconds, the original time series will be divided into 30-second segments and a weighted average will be calculated for each segment. The resulting file will still contain an onset column and one or more data columns; however, the onset column will now contain only multiples of 30 and the values in each data column will correspond to those 30-second weighted averages. The resulting files are saved within a new folder, created as a subfolder of the input folder.

If the file length itself is not an exact multiple of the bin size you select, the last bin in the data set will be incomplete. For example, if we are using a 140-second time series and bin size of 30 seconds, we will have four full bins (spanning 0 seconds to 120 seconds) and one incomplete bin, which will only be 20 seconds. The input sheet of this macro provides you with the option of including or excluding this incomplete bin. If you choose to include it, a weighted average will be calculated across whatever data is present (i.e., across the 20 seconds leftover). If you choose to exclude it, all data in the final bin will be deleted, and final data point in the resulting binned file will correspond to the last *full* bin. A third option is to enter an integer in Cell C7 and set a threshold, where the last bin will only be included if it is a certain percentage of a full bin. For example, if you enter “50” in this cell, the macro will only include the final bin if it is at least 50% of the size of a full bin. In our previous example, our final 20-second bin *would* be included, as it is more than 50% of the full bin size (30 seconds).

**Assumptions:** All files within the input folder must be text/.trj files that you wish to transform into binned data files. The input files must also contain no empty columns, other than those to the right of all the data. Finally, the input files must contain an onset column with a final value denoting the end of the timeframe.

**Input:**  


**Example of One Original File: Resulting File:**  


## 3.11) Categorization

Categorizes continuous time series data into intervals of equal width to allow for later categorical analysis (generating State Space Grids, for example). The user specifies the desired number of categories they wish to create for each variable. In order to create these categories, the macro must first calculate the maximum and minimum value of each variable, then determine the difference and divide by the desired number of categories. In order to determine the maximum and minimum values, the user has two options: to calculate within participants or across participants.

If the user selects across participants (by entering “Y” in Cell E29), the first step of the macro is to loop through all files in the input folder and find the overall, sample-wide maximum and minimum of each variable. The final max and min values are entered into Sheet2 in the macro workbook for future reference. The macro then uses these overall max and min values to calculate the width of each category (subtracts the min from the max and divides by the user-specified number of categories). Once these categories are calculated, the macro loops through each file a second time and assigns each data point to the appropriate category. The category value is expressed as a range and is pasted into the corresponding row within the first available column. Note that since the max, min and category width are calculated first, this option uses the same categories for each file, making the resulting categorized data more comparable across participants.

If the user selects within participants (by entering “N” in Cell E29), the maximum and minimum value for each variable will be different for each participant. Rather than first looping through each file to find the *overall* max and min values, the macro opens each file just once, setting the maximum and minimum to a new value each time for each file. The macro then uses these participant-specific values to calculate the width of each category (subtracts the min from the max and divides by the user-specified number of categories). Finally, the macro uses these participant-specific categories to assign each data point. The category value is expressed as a range and pasted into the corresponding row within the first available column. The macro then moves on to the next participant and repeats all of the above steps. Note that since *all* steps (i.e., calculating the min and max, finding the category width, assigning the data points) are performed for one participant before proceeding to the next, the min, max and category width will be different for each participant. The number of categories, however, will remain constant, thus each participant will have the same number of categories but those categories will span a different range of values. One way to make these categories comparable across participants is to use standardized variables as input. Then, catgories would represent within-subject standard deviations.

As mentioned, the resulting category is pasted as a range, showing the minimum and maximum as a string. For example, if the data point in the original variable column is 12.34, and one of the calculated categories ranges from 10.05 to 15.75, the matching row within the “Categorized” column will read “10.05 to 15.75”.

The user also has the option of including a column of category “labels” for each column of categorized data. If the user selects “Y” (in Cell E30), the macro will generate an integer value for each category, where 1 corresponds to the first or lowest interval (i.e., includes the minimum), 2 corresponds to the next interval, etc. The appropriate integer value will be pasted next to the categorized column. For example, if the category “10.05 to 15.75” was the lowest interval (i.e., 10.05 was the minimum), the corresponding rows in the adjacent column would be assigned a “1”. Similarly, if the category “10.05 to 15.75” was the highest interval (i.e., 15.75 was the maximum), and there were a total of 10 categories, the corresponding rows in the adjacent column would be assigned a “10”.

A new data file is generated for each participant and saved within the “Desired Output” folder, which will be created as a subfolder within the original input folder.

**IMPORTANT:** Do NOT rename the macro workbook or the names of Sheet1 and Sheet2 as the code itself would also have to be altered to recognize the new names.

**Assumptions:**

* All column numbers must be specified in the input sheet using integers.
* All original data (in the specified columns within the input files) must be in percent change format, but can be in decimal (e.g., .54) or percentage (e.g., 54%) form or standardized (e.g., z-score). If this is not the case, run the “Baseline Deviations” macro ([Section 5.1](#_5.1)_Baseline_Deviations)) prior to using this macro (to get decimals or percentages) or standardize your existing data. The user must specify whether they are working with decimals, percentages or z-scores in Cell E15. All columns you wish to categorize must be in the same format (i.e., cannot have some columns in decimal form and others in percentage form).
* In all original files, data should begin in Row 2 of each variable column (i.e., it is assumed that Row 1 is occupied by the column headers).
* Input files must be a series of files (one per participant) within a single folder.

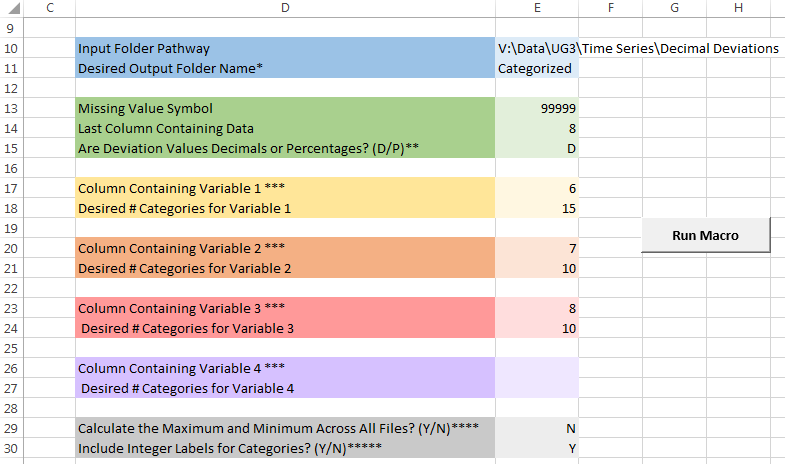
**Options:**

* Can categorize up to four variables.
* Can work with decimal or percent deviations (Cell E15). If working with percentages, maximum, minimum and category values are calculated directly. If working with decimals, maximum and minimum values are multiplied by 100 to create categories that show a range of *percentages*. All data points in the original column(s) are also multiplied by 100 to ensure they are assigned to the proper categories (e.g., 0.0567 becomes 5.67%). When the column data is converted from decimals to percentages, the column header is modified to include “\_%” to avoid any later confusion when looking at the finished files. **In both cases, resulting categories show percent change,** and may be positive or negative (above or below the baseline value).
* Can calculate **one** overall maximum and minimum across all files (categories will be identical for all participants) or calculate individual maximum and minimum values for each file (categories will be different for each participant). This Yes or No option is provided in Cell E29.
* Can include integer values as category labels, where an integer is pasted into the column adjacent to the categorized column, indicating where that category falls with respect to the other categories. For example, an integer of “1” indicates that the category in the previous column is the *first* category (i.e., includes the minimum). This Yes or No option is provided in Cell E30.

Two examples are illustrated below to show the various options. Note that the images of the resulting files are cut off – in practice, the final files will include **all** of the original data; the categorized columns are added to the first free columns on the right of the original data.

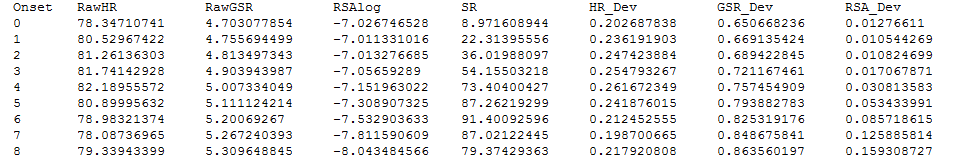
**Example 1: Decimal Form, Within-Participant Calculations, Labels Included**

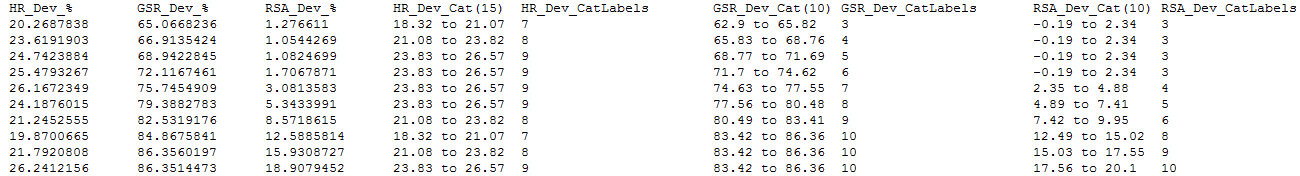
**Input:**



Note that the input in Cells E15, E29 and E30 correspond to our preferences for this example (working with decimals, calculating within-participant maximum and minimum values, including integer labels).

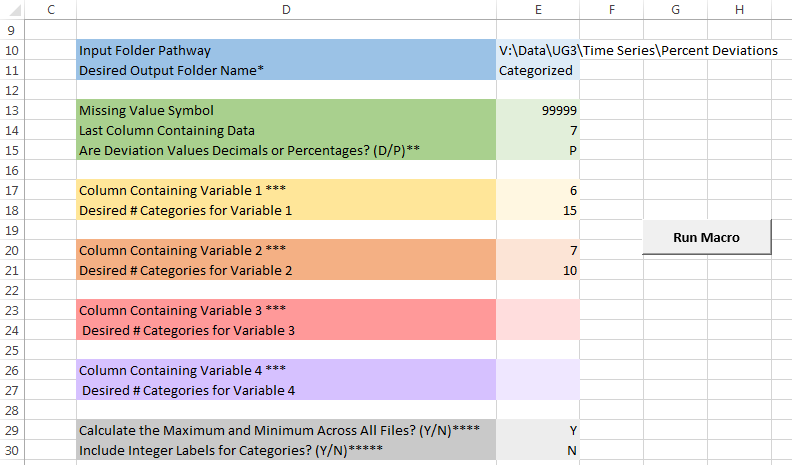
**Example of One Original File:**



**Resulting Categorized File:**  


Note that “\_%” has been added to the original data columns (HR\_Dev, GSR\_Dev, etc.) to clarify that the decimal values have been changed to percentages. The categorized columns are automatically labeled using” \_Cat(X)” as an extension on the original data column heading, where X is the number of categories used for that variable. The column containing integer labels is automatically labeled using “\_CatLabels” as an extension on the original data column heading. **Important:** Since we selected within-participant max/min calculations, the categories for the other participants would be **different!**

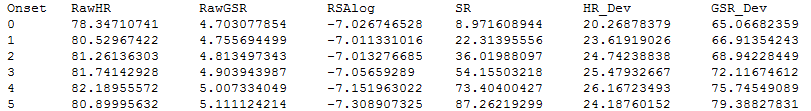
**Example 2: Percentage Form, Across-Participant Calculations, No Labels**

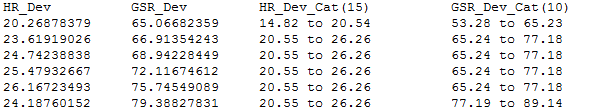
**Input:**

Note that the input in Cells E15, E29 and E30 correspond to our preferences for this example (working with percentages, calculating across-participant maximum and minimum values, excluding integer labels).

**Output to Sheet2 (Overall Max/Min Values):**  


Note that this output only occurs if using across-participant max/min calculations. Also note that if we were working with decimal data, these values would be in decimal form but would be multiplied by 100 in the code before continuing the macro.

**Example of One Original File:**

**Resulting Categorized File:**

Note that the headings on the original data columns (HR\_Dev and GSR\_Dev were not changed this time because they were already in percentage form. Also note that since we entered “N” for “Include integer labels” there is only one new categorized column for each variable.

# 4) APPENDING TIME SERIES DATA

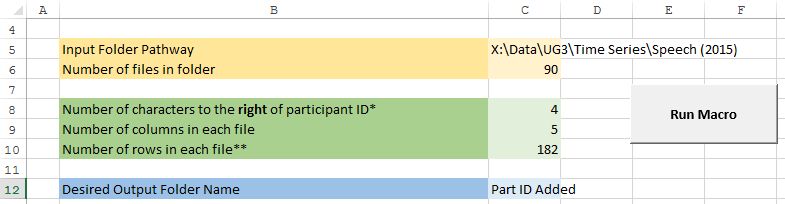
## 4.1) Remove Empty Row

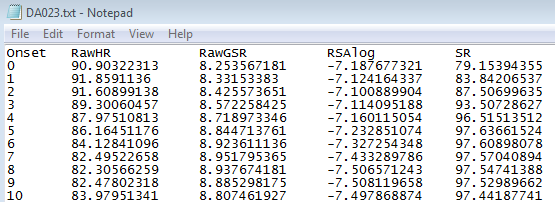
This macro will clear any data in and beyond a user specified row. This is helpful when dealing with onset-corrected files, where the last row contains no data. Note that you must enter the row **number,** not the onset value. For example, in a 180-second time series file, the last row corresponds to an onset time of 180 but is actually row number 182 (due to the header in Row 1 and Onset = 0 in Row 2).  
It is not necessary to run this macro before running the append macros (“Append Full Time Series” [[4.4]](#_4.4)_Append_Full) or “Stacking Variables and Dummy Coding” [[4.5]](#_4.5)_Stacking_Variables)) as these macros either automatically exclude the last row (4.5) or provide you with the option of including or excluding it (4.4).

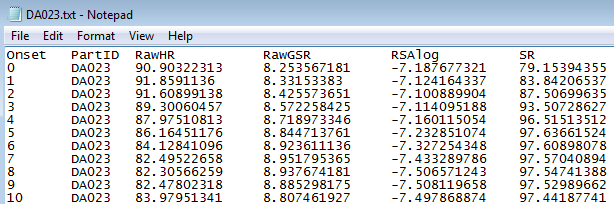
## 4.2) Adding Participant ID Column

This macro isolates the Participant ID for each file and creates a PartID column. Onset values remain in Column 1, PartID is inserted in Column 2, and all other columns are shifted one column to the right. The macro loops through all files within the input folder and inserts the appropriate PartID for each one.

**Assumptions:** Participant ID must be the first characters of filename. If this is not the case, run the “Removing Filename Beginning” macro ([Section 2.2](#_2.2)_Removing_Filename_1)) first.

**Input:**

**Example of One Original File:**

**Resulting File:**  


## 4.3) Adding Columns of Variables with Constant Values

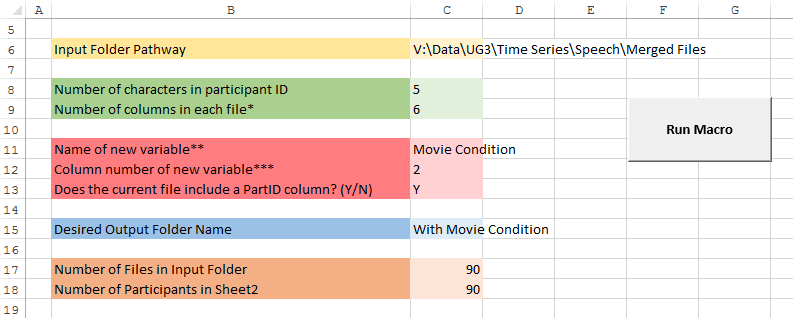
This macro inserts a time-invariant variable (i.e. a variables whose value does not change throughout the time series [age, sex, etc.]). Similar to the “Adding Participant ID Column” macro ([Section 4.2](#_4.2)_Adding_Participant)), each new variable is inserted as a new column within the existing merged file. Onset and Participant ID are left in columns 1 and 2. All new variables are inserted after Participant ID and before the physio time series data. **All variables have to be added one-by-one.**

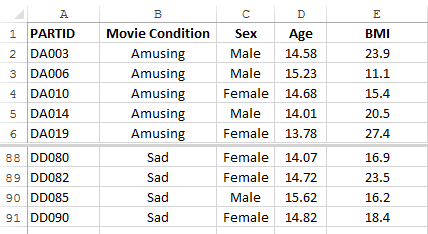
Data for all variables of interest must be entered in Sheet2 of the worksheet. The macro will match the Participant ID in the filename to the Participant ID in column A of Sheet2 and input the corresponding variable value. Note that only one variable can be inserted at a time, and that cells C9, C11 and C12 must be changed each time. If a participant is missing the specified variable, missing values will be inserted for the full column in the new file. Similarly, if a participant has a merged file containing physio data but does not have any data input into Sheet2 of the macro, missing values will be inserted for all invariant columns.

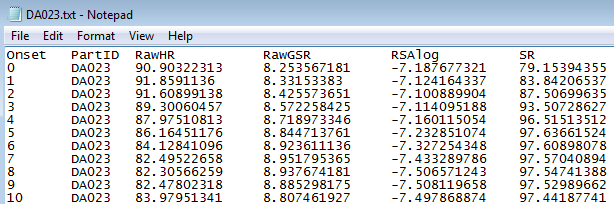
**Note:** If you wish to add a new variable to a specific column, you must enter it into the appropriate column of Sheet 2.

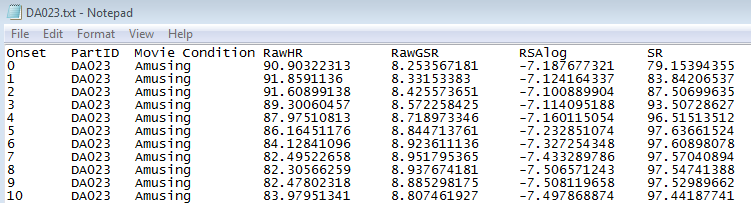
If input files **do not** have a Participant ID column: Column number in Sheet2 should be the same as the desired column number in your new files (i.e. if you want the new variable in column 5, type it into column 5 (column E) of Sheet2.

If input files **do** have a Participant ID column: Column number in Sheet2 should be **one less than** the desired column number in your new files (i.e. if you want the new variable in column 5, type it into column 4 (column D) of Sheet2.

**Input:**

**Input (Sheet2):**

**Example of One Original File:**

**Resulting File:**

## 4.4) Append Full Time Series

Creates an appended file for data containing time series variables (i.e. heart rate, GSR) as well as invariant variables (i.e. age, sex, BMI). The time series variables will be different for each time point, while the invariant data will be constant across the entire time series length for each participant. For example, if the time series is 180s long, the participant ID and invariant variable data is repeated for all 180 rows, while the onset and time series values will be different for each row. This appended format is useful for many types of analysis (e.g., multilevel modeling).

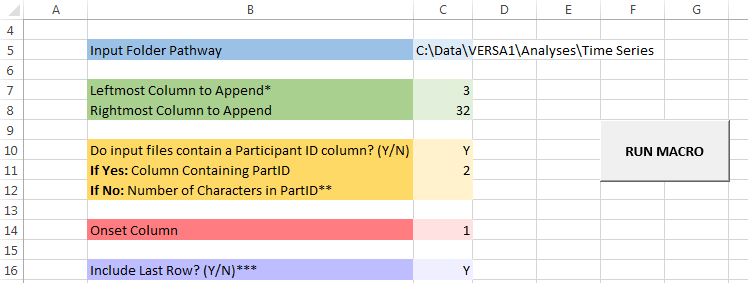
The input files for this macro will be a series of merged text files (specified by the input folder). The macro loops through all of the text files in the folder, copying and pasting the data from each one into a single Excel file. The resulting Excel spreadsheet will contain several thousand rows, as each participant will have one row for each second of the time series. As the macro cycles through each file, the data for each new participant is inserted below the previous one. For example, if the time series is 180 seconds long, the resulting file will begin with 180 rows for participant AB001, followed by 180 rows for participant AB002, and continue until all participant data is pasted to the file.

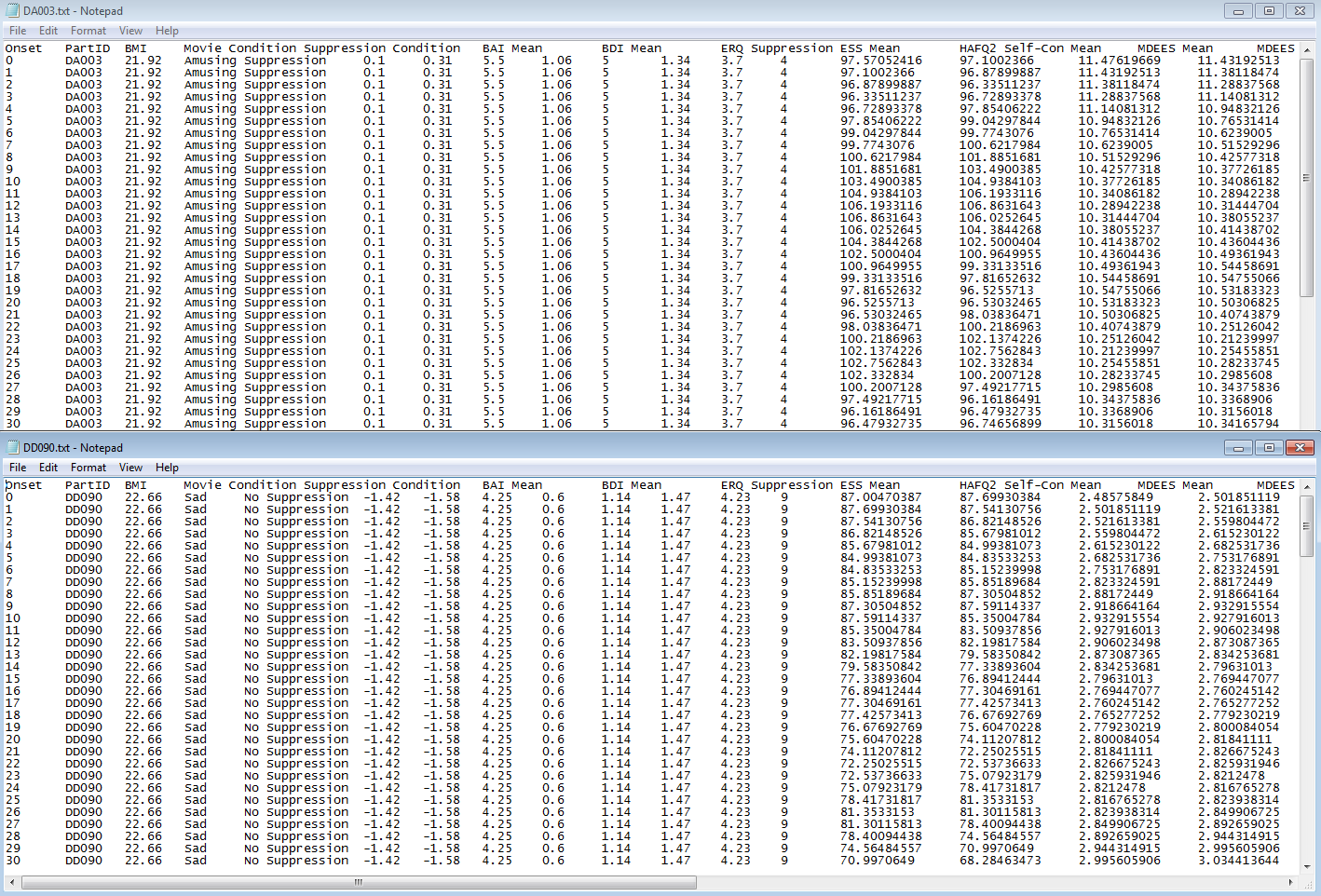
Since the last row of a time series generally contains no data (due to the shift when correcting the onset or when using the final offset files for GridWare), this macro gives you the option to include or exclude the last row. If you enter “N” in Cell C16, the macro does **not** copy and paste the last row. For example, in a 180-second time series, each participant would have 180 rows of data ranging from 0 seconds to 179 seconds. If you enter “Y” in Cell C16, the macro **does** copy and paste the last row. This is useful when working with binned data, for example, or any files for which the last row is not empty.

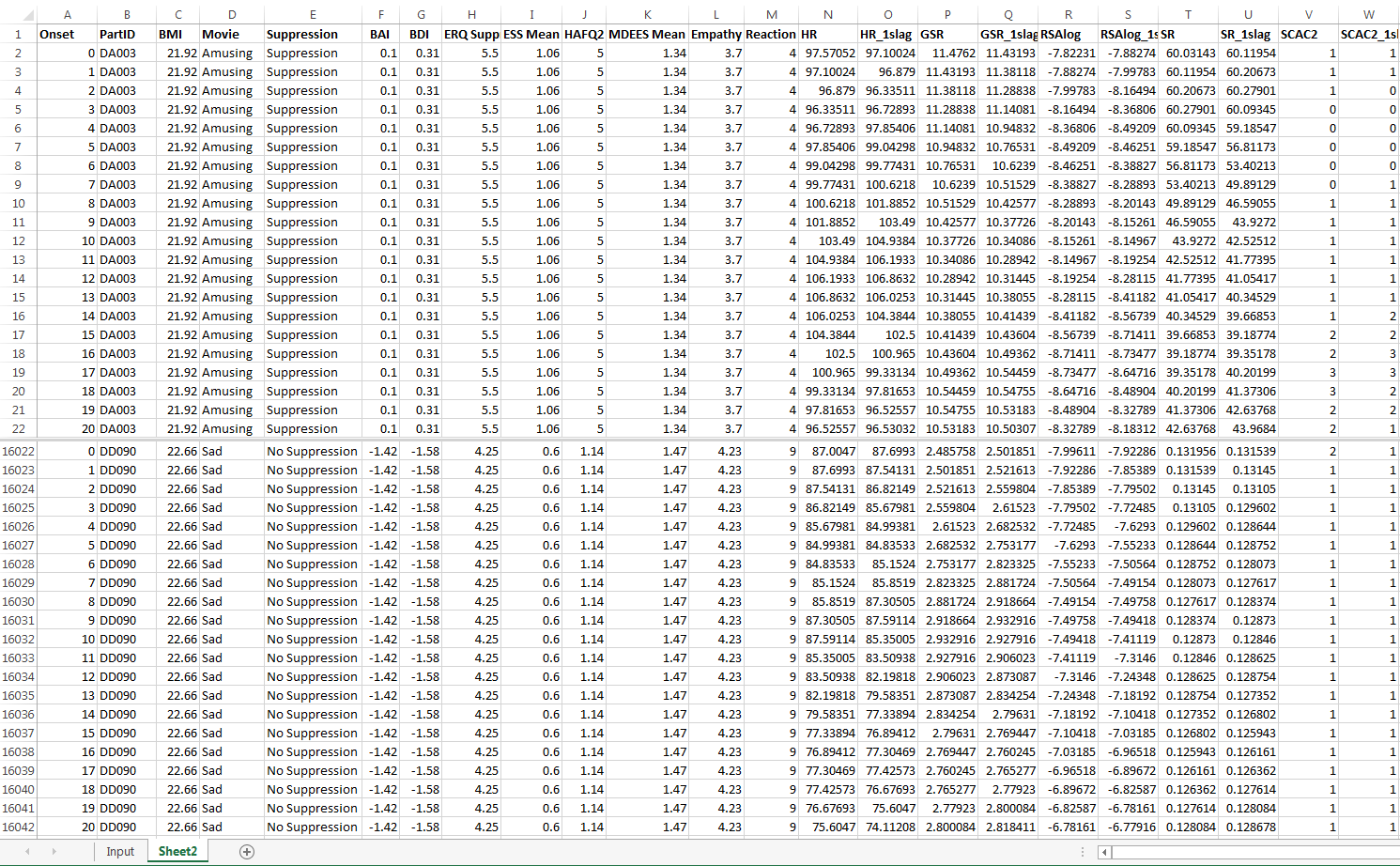
The resulting Excel spreadsheet will be pasted to Sheet2 of the “Append Full Time Series” macro. **You must save the entire Excel file in the desired location using the “Save As” button.** Do not just save the macro, as your data in Sheet2 will be overwritten the next time the macro is used.

**Assumptions:** The option to include or exclude the last row of data will be applied to all files in the input folder (i.e., cannot include it for some files and exclude it for others). Input files do not have to be the same length. If input files do not have a participant ID folder, a participant ID column will be created using the filename, therefore ensure the filename **begins** with the participant ID (if it does not, run the “Removing Filename Beginning” macro ([Section 2.2](#_2.2)_Removing_Filename_1)).

**Input:**



**Example of Two Separate Files:**

**Resulting Appended File:**

## 4.5) Stacking Variables and Dummy Coding (for SPSS Mixed Models and Multilevel Modeling)

This macro also creates a fully appended Excel file by combining the time series data of all participants into a single spreadsheet. In this stacked format, all time series (physio) data is pasted into a single column (the Z column). Each variable in the data file is appended to the bottom of the previous variable. The macro then dummy codes the resulting spreadsheet to indicate which variable is represented at each time point.

Similar to the “Append Full Time Series” macro ([Section 4.4](#_4.4)_Append_Full)), the input files will be a series of files within a single folder. The macro loops through this folder and stacks the data for each participant. The resulting file is a single Excel spreadsheet, located in “Sheet2”. Since the time series data is being stacked this time, rather than pasted into separate columns, this appended file will be very large (up to several hundred thousand rows, depending on the number of variables and the number of participants).

This macro also gives you the option of including invariant variable data (in which case input files must include constant value columns, obtained using the “Adding Columns of Variables with Constant Values” macro, [Section 4.3](#_4.3)__Adding)). These values will be inserted after the participant ID column and before the Z-column containing all of the time series values. When you are **not** including any invariant variables, all cells below Cell C25 may be left blank, and the “Run Time Series Append” button should be used. When you **are** including invariant variables, **all cells\*** must be filled out and the “Run Invariant Append” button should be used.

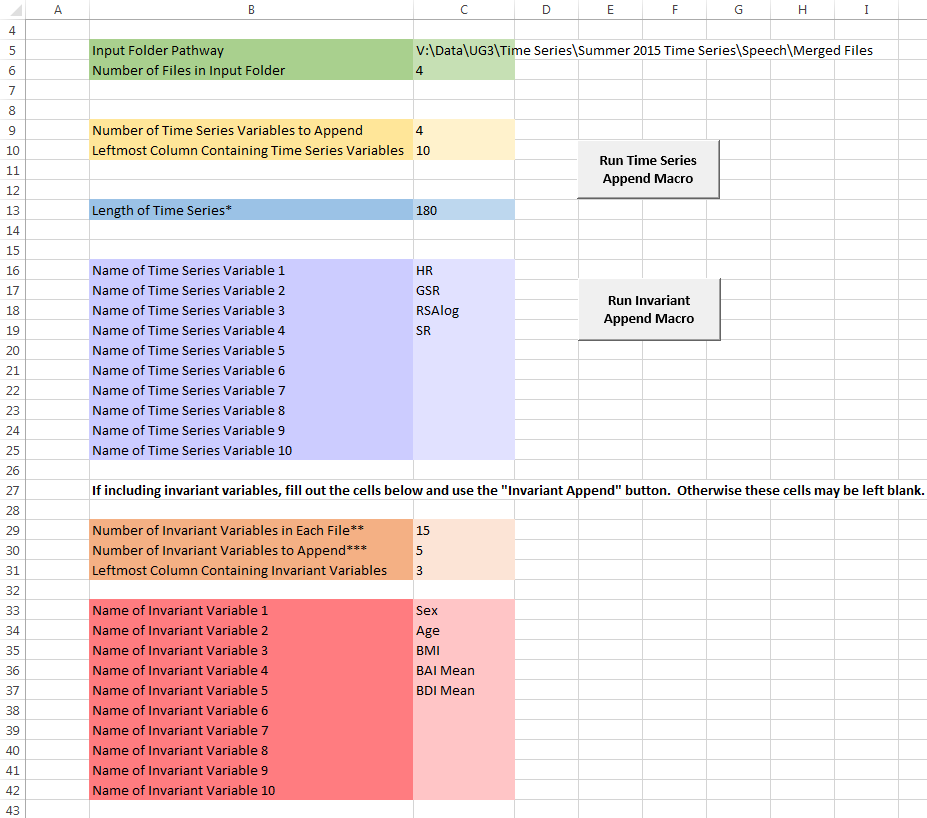
**Assumptions:**

* All files within the input folder must have Onset in column 1 and Participant ID in column 2.
* The macro is currently equipped to append a maximum of 10 time series variables and 10 invariant variables. Relatively simple adaptations can be made to the code in order to accommodate a larger number of variables. These areas and required changes are highlighted in the comments within the code.
* The macro is currently designed to exclude the last row of each file, as it assumes you are working with onset-corrected files where the last row is empty. Again, simple adaptations can be made if you wish to include the last row; these areas are highlighted in the code comments.
* When including invariant variables, the user must specify how many exist in the original files, as well as how many they wish to include in the appended file. This information is important to the dummy coding of the time series variables. In order for the dummy coding to function properly, the user cannot exclude more than 10 invariant variables from the original files. For example, if you are using merged files with 20 invariant variables, you must be including at least 10 of them in the appended file. Again, adaptations can be made to the code if you wish to exclude more than 10 variables, and these areas are highlighted in the comments.
* When including invariant variables, they must be side-by-side in the original files as they are copied and pasted as a single range and not as individual columns.

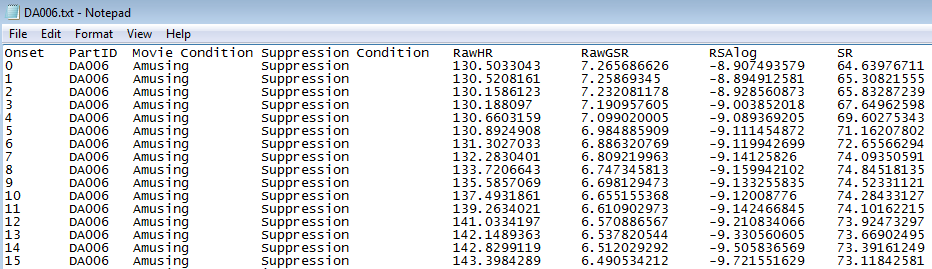
**IMPORTANT:** The resulting appended file will be pasted to Sheet2 of the “Stacking Variables and Dummy Coding” macro workbook. **You must save the entire workbook in a desired location using the Save As function.** If you just hit “Save”, your data will be overwritten the next time the macro is used.

**\***If you are using fewer than 10 time series or invariant variables, cells corresponding to variable names may be left blank. These cells are used for naming the column headers in the resulting appended file.

**Input:**

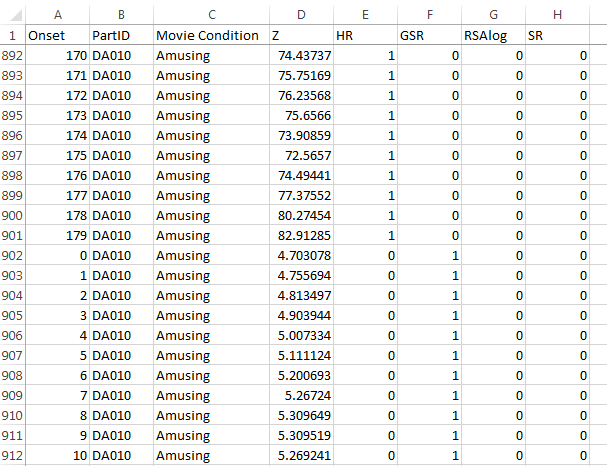


**Example of One Original File:**



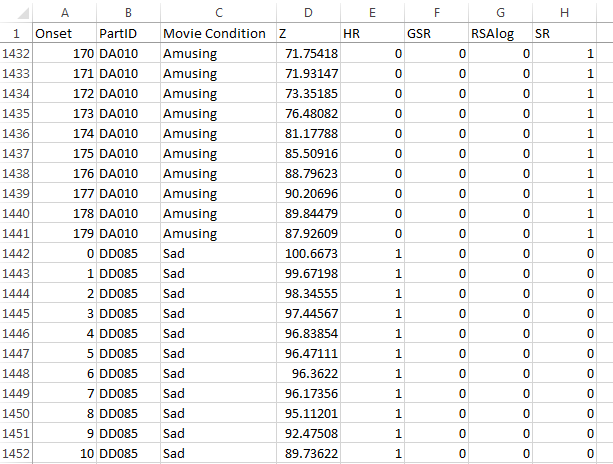
Note that this input file has two invariant variables (Movie Condition and Suppression Condition) but we are only including one of them, as specified in the input sheet. Since we specified “3” as the leftmost column, we will be including Movie Condition.

**Resulting Appended File (Image 1):**



This image shows the switch between one participant’s HR and GSR data. The onset continues all the way to 179 (the 180s row is excluded) and then begins again at 0. The invariant variable (Movie Condition) is repeated for all rows corresponding to the participant. In the dummy coded columns, there can be only one “1” for each row, with all other values at “0”. Note that the 1’s are in the HR column until row 182, at which point the data in the Z-column switches from HR data to GSR data. At this point, the values in the HR column become 0’s and the values in the GSR column become 1’s. This switching pattern continues for all variables, then switched back to 1’s in the HR column when the next participant is added.

**Resulting Appended File (Image 2):**



This image shows the switch between one participant and the next. Note that the PartID and invariant variable data now changes. The Z-column begins again with the HR data and is dummy coded accordingly, switching from 1’s in the SR column (for the last few rows of participant DA010) to 1’s in the HR column (for the first few rows of participant DD085).

# 5) CREATING NEW VARIABLES

## 5.1) Baseline Deviations

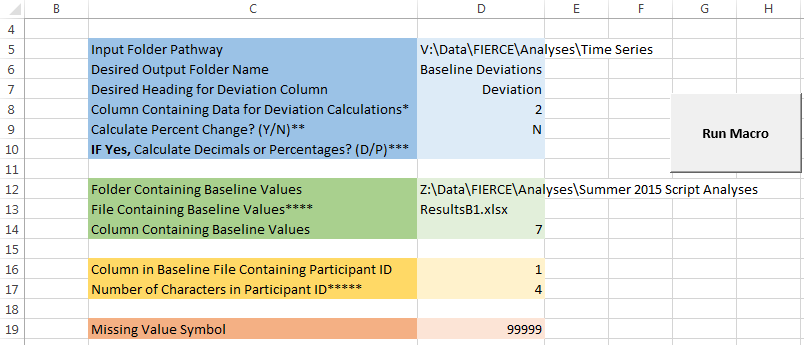
Compares an entire time series to a single average baseline value. For example, user can compare each second-by-second HR value during a speech task to the average HR value during a baseline task. The deviations are pasted to the first available column within the time series file, creating a “time series” of second-by-second deviations.

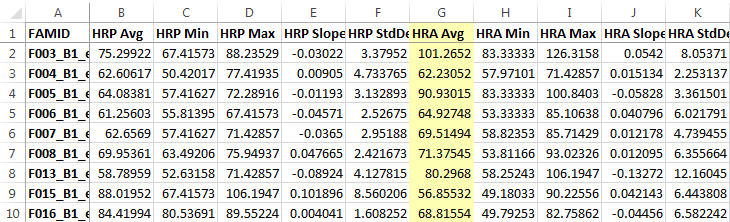
The user has the option of calculating direct deviations, where the deviation value is just the difference between the raw score and the average, or percent change deviations, where the deviation value is the percent difference between the raw score and the average. If calculating percent change, user also has the option of obtaining final results as decimals or as percentages by entering “D” or “P” in Cell D10.

If the baseline value is a missing value (e.g. 99999) and/or the value(s) in the time series is a missing value, the deviation value will automatically be inserted as a missing value.

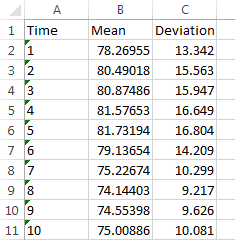
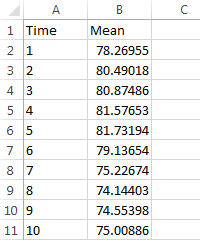
**Assumptions:**

* Participant ID must be the first few characters of the filename. If this is not the case, run the “Remove Filename Beginning” macro ([Section 2.2](#_2.2)_Removing_Filename)) before using this macro.
* If a participant has a time series file but there is no row in the baseline file to match that participant ID, the deviation column will be blank for that participant. However, the column header for the deviation column will be added, therefore if you are unsure whether or not some participants are missing from the baseline file, you can simply run the “Missing Value Insertion” macro ([Section 3.4](#_3.4)_Missing_Value)) on the new deviation files.
  + **Note:** An alternative solution (when working with relatively small datasets) is to ensure all Participant ID’s are entered into the Participant ID column and that their values are replaced with missing values. In this case, the entire Deviations column will automatically be filled with missing values.

**Input:**

**Example of Baseline File:**

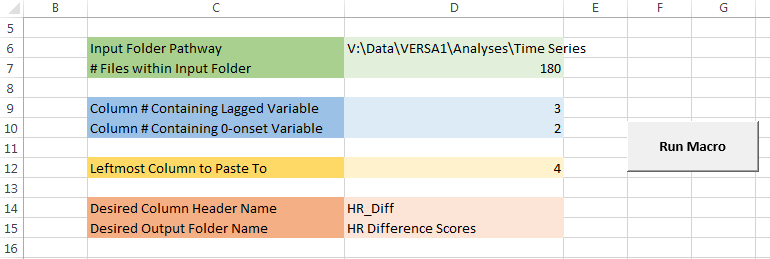
In this example, we are calculating the deviations for the adolescent heart rate, therefore we are using Column G (Column 7) as the baseline column.

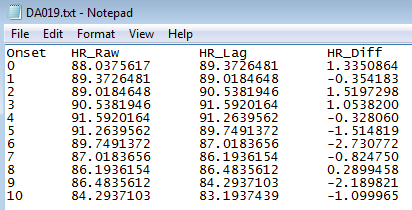
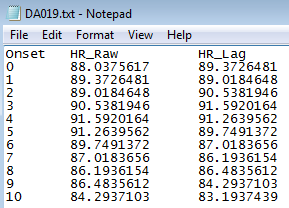
**Example of One Original Input File: Resulting File:**

Note that these deviation values are direct deviations (because we input “N” for percent change in Cell D9). Thus, the first deviation value in Column C of the resulting file indicates that this participant’s heart rate at a time of 1 second was 13.342 beats per minute higher than their baseline heart rate.

## 5.2) Difference Scores

Creates a series of within-subject difference scores by subtracting the lagged variable at each second from the 0-onset variable at that second. In order to use this macro, user must first create the lagged variable and add it to the 0-onset time series file (using “Lagged Variable Creation”, [Section 5.4](#_5.4)_Lagged_Variable)).

**Input:**

**Example of One Original File: Resulting File:**

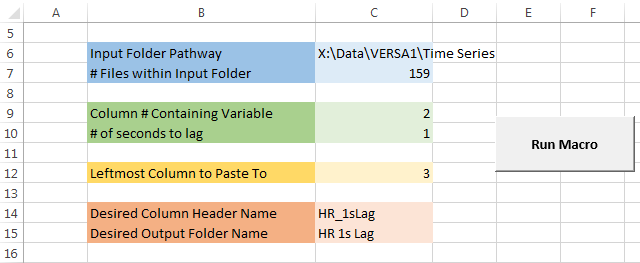
## 5.3) Logged Variable Creation

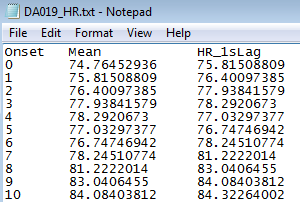
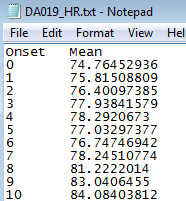
Creates a new variable column by taking the natural log of all data in a specified column or range. Designed for the purpose of obtaining more useful information from RSA time series data. Refer to [Section 3.3](#_3.3)_Logged_Variable) for description and examples.

## 

## 5.4) Lagged Variable Creation

Creates lagged time series by offsetting the data in a specified column by a specified number of seconds.

**Input:**

**Example of One Original File: Resulting File:**

# 6) GRAPHING

## 6.1) Single Channel with Dynamic Axes and Task Coding

Creates a smooth scatterplot using onset along the x-axis and a user-specified variable along the y-axis. User does **not** indicate a desired scale for the axes. Instead, axes will be formatted to fit the minimum and maximum values of each participant. The resulting graphs are copied and pasted into a Word document and will have **different** x- and/or y-axis scales for each participant.

This macro also asks user to indicate whether or not the onset column is being used as either the x- or y-axis variable. This information is used when formatting the axes. If the axis variable is onset, all tick values along the axis are rounded to zero decimal places. If the axis variable is **not** onset, all tick values are rounded to three decimal places. If onset is **not** one of the axis variables, time is embedded within each data point. For example, if graphing GSR against RSA (as in Example 2, below), one data point represents an (x,y)-coordinate that corresponds to (GSRvalue, RSAvalue). Each point is connected to the next point in the series (i.e. the next second) by a line.

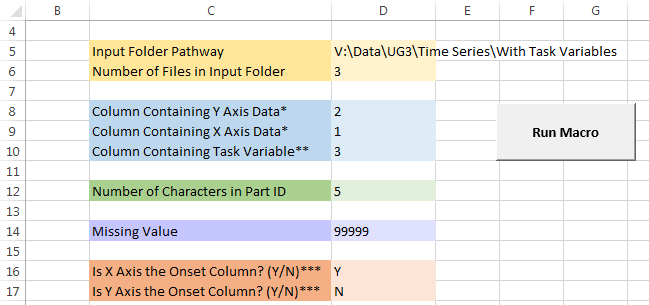
This macro also gives users the option to colour-code the graph using the previously mentioned task variables. If this option is selected, the resulting graph will included segments of different colours, corresponding to the different parts of the task. For example, the points corresponding to Task 1 will be one colour, and the points corresponding to Task 2 will be a different colour. This macro is able to code up to 5 different colours in a single graph.

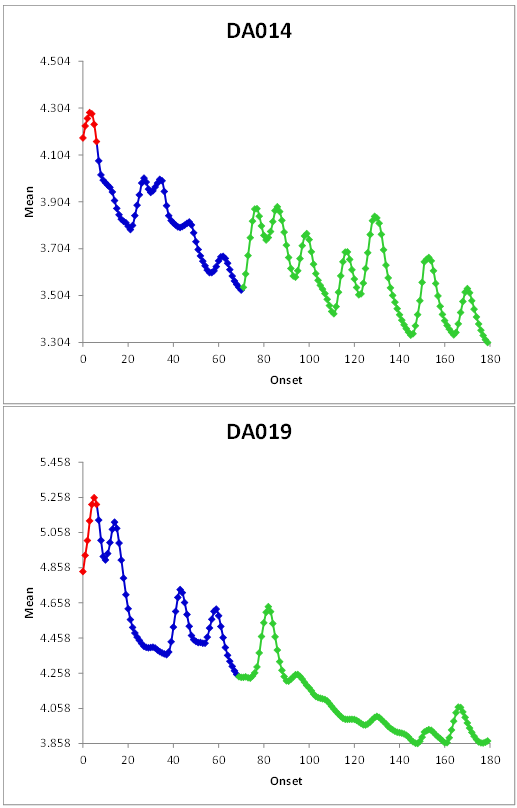
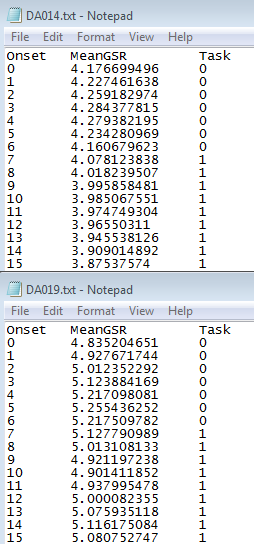
**Note:** In order to use the colour-coding feature, the “Adding Task Variables” macro ([Section 3.1](#_3.1)_Adding_Task)) must be run first.

**Assumptions:** The two variables you wish to graph must be consistently in the same columns throughout all files in the input folder (i.e., cannot have some files where your onset variable is in column 1 and some files where it is in column 2). All column values must be supplied as integers.

The two examples below illustrate the various options the user has when running this macro.

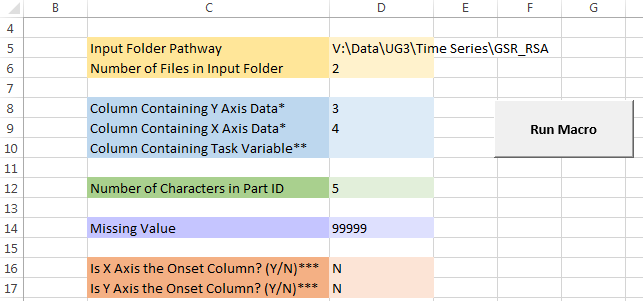
**Example 1: Including Onset and Task Variables**

**Input:**Note that the user does NOT specify the maximum or minimum values for the x- or y-axes. In this example, we **are** using onset for one of the axes (Cell D16) and we **are** including a task variable (Cell D10), therefore our graphs will be colour-coded according to those task variable values.

**Example of Two Original Files: Resulting Graphs:**

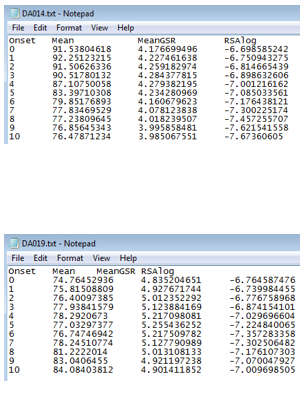
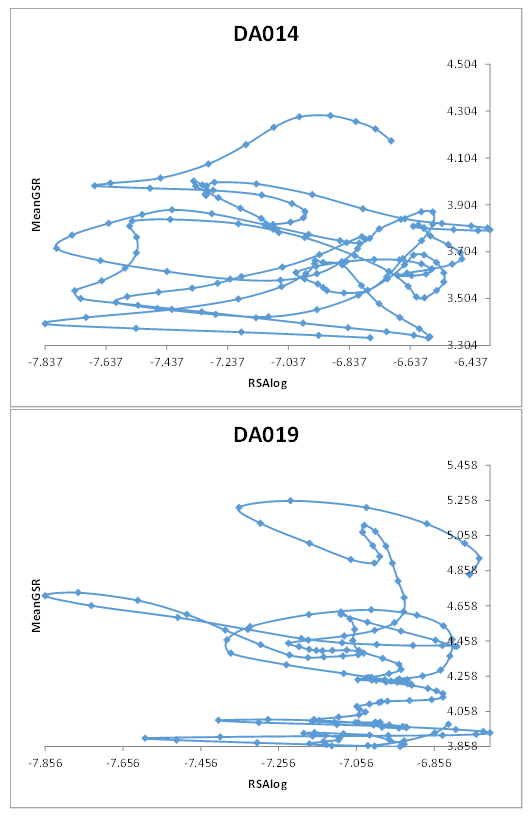
**Example 2: Excluding Onset and Task Variables**

**Input:**



Note that the user does NOT specify the maximum or minimum values for the x- or y-axes. In this example, we **are not** using onset for either of the axes (Cells D16 and D17) and we **are not** including a task variable (Cell D10), therefore our graphs will not be colour-coded and both axes will be rounded to three decimal places.

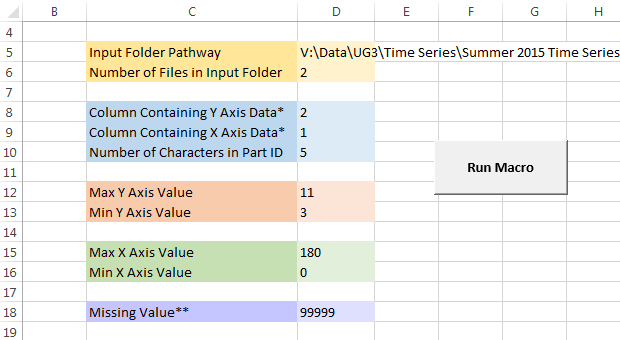
**Example of Two Original Files: Resulting Graphs:**

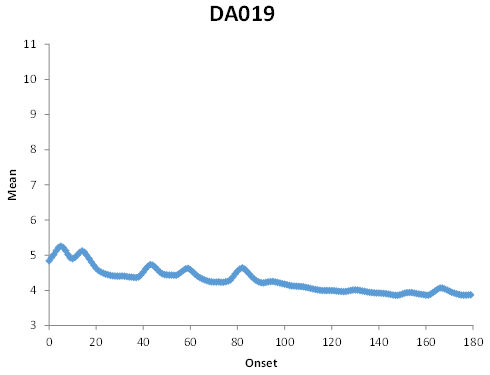
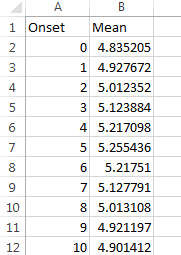
 

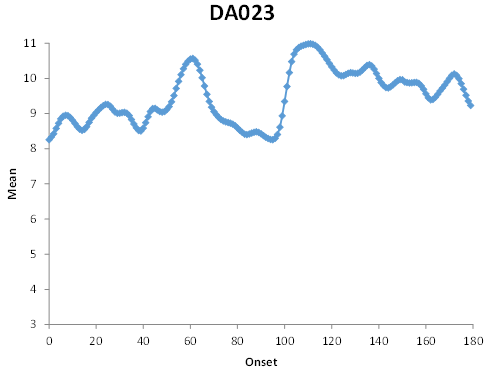
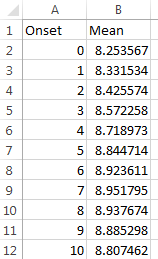
Note that the x- and y-axes are different for each participant, and that the tick values are rounded to 3 places.

## 6.2) Single Channel with Specified Axes Values

Creates a smooth scatterplot using onset along the x-axis and a user-specified variable along the y-axis. User can customize the axes by inputting their desired minimum and maximum values. The resulting graphs are copied and pasted into a Word document, and will have the same scales for all participants.

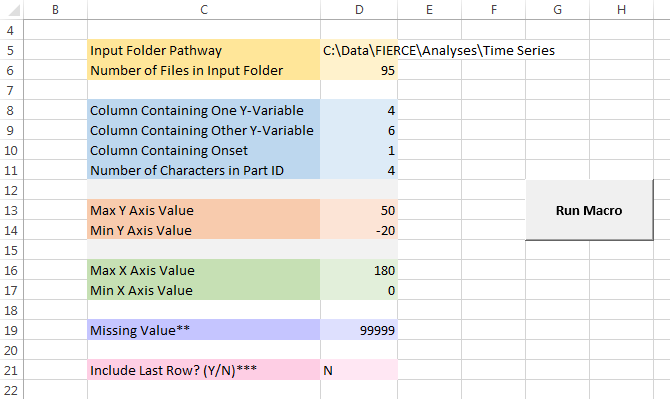
**Input:**

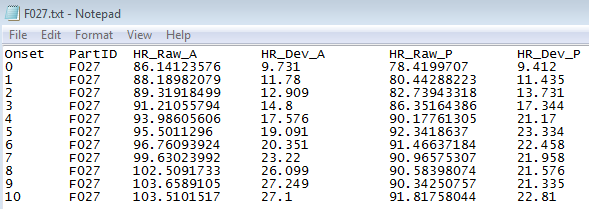
**Example of Two Original Files: Resulting Graphs:**

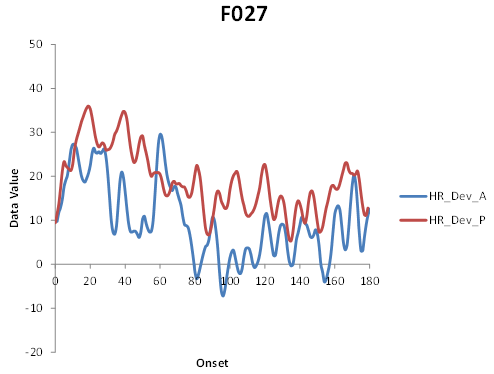
 **Note** that both graphs are not colour-coded and that the x- and y-axes are the same for each participant.  
  
**Important:** As with [Section 6.1](#_6.1)_Single_Channel), the x-axis does not have to be the onset column. However, due to the major variation in participants for other variables (e.g., heart rate, GSR), graphing two physio variables with fixed axis values can make the resulting graphs very difficult to interpret.

## 6.3) Two Channels with Specified Axes Values

This macro graphs two time series variables (e.g. RSA and GSR) along the y-axis against “onset” along the x-axis. The range of both axes is specified by the user and will therefore be the same for each graph. The legend titles on each graph are taken from the column headings of each corresponding variable. The graphs are pasted into a single word document. The last row of data can be excluded if it contains missing values, as this may cause a sudden spike at the end of the graph.

**Input:**

**Example of One Original File:**

**Resulting Graph:**

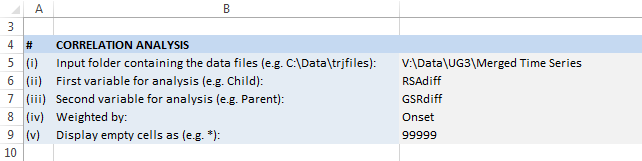
# 7) CORRELATIONS

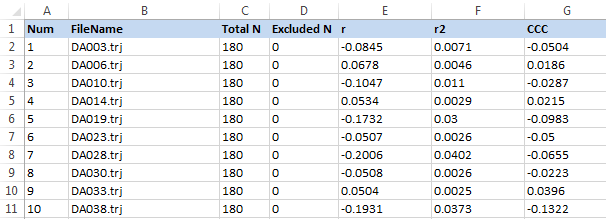
## 7.1) Within-Subject Zero-Order Correlations

Creates within-subject zero-order correlations between two user-specified columns within a merged file. The correlation *(r),* variance *(R2)*, and concordance correlation coefficient\* *(CCC)* values are pasted into the “results” sheet of the macro workbook. The macro will loop through all merged files within the input folder; the “results” sheet will contain one row of values for each participant.

**Note:** The two columns you wish to correlate must be within a single file. If this is not the case, run the “Customized Merging” macro ([Section 3.2](#_3.2)_Customized_Merging)) first. Also note that unlike other macros, this macro requires you to identify the correlation columns using their titles, not column numbers.

**Important:** Do not rename the sheets within this macro workbook (i.e. “correlation” and “results”) as it will interfere with the code. To save your correlation results, **save the entire workbook using the Save As function** or copy and paste the data from the “results” sheet into a new workbook.

**Input:**  


**“Results” Output Sheet:**

\*The concordance correlation coefficient (CCC) is calculated using the following equation:

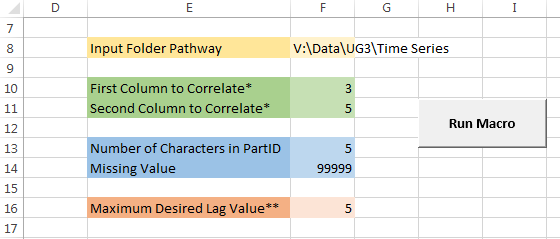
## 7.2) Customizable Lagged Correlations

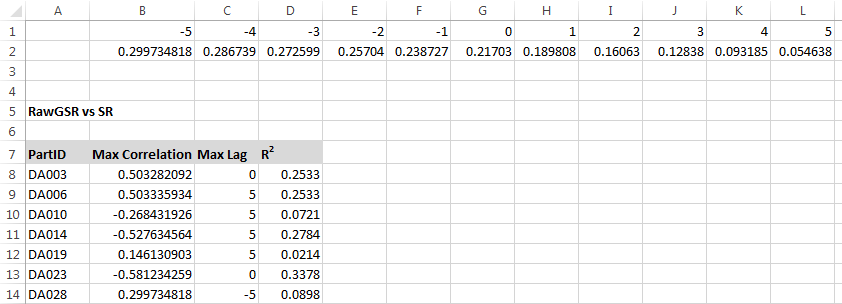
Calculates multiple correlations using a time lag specified by the user, then selects the largest value to paste to the output sheet. The lag time to which this maximum value corresponds is pasted in the adjacent cell. The chosen correlation value is then used to calculate an R2 value, which is pasted to the next available column. The results are pasted to Sheet2 of the macro workbook (the “Output” sheet).

For example, if the user specifies a lag of 5 seconds, the macro generates a total of 11 correlation coefficients for each participant: positive lag (1 to 5 seconds), no lag (zero seconds), and negative lag (-1 to -5 seconds). The macro then determines which of the 11 values is the largest and pastes it into the output sheet with its corresponding lag time. For example, if the largest correlation was 0.5678 and was calculated at a lag of +3 seconds, then “0.5678” is pasted to the output sheet and “3” is pasted in the adjacent cell.

**Note:** The two columns you wish to correlate must be within a single file. If this is not the case, run the “Customized Merging” macro ([Section 3.2](#_3.2)_Customized_Merging)) first.

**Important:** Do not rename the sheets within this macro workbook (i.e. “UserInput” and “Output”) as it will interfere with the code. To save your correlation results, save the entire workbook using the **Save As** function, or copy and paste the data from the “Output” sheet into a new workbook.

**Input:**

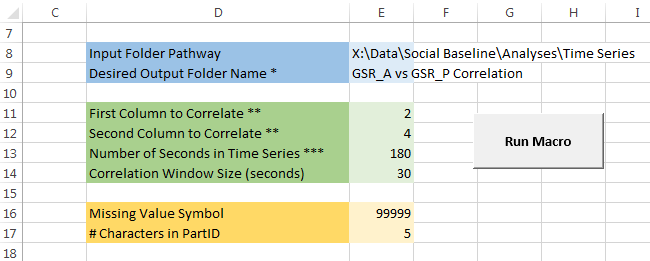
**Resulting Output Sheet:**

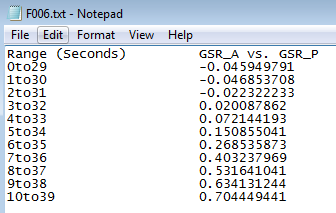
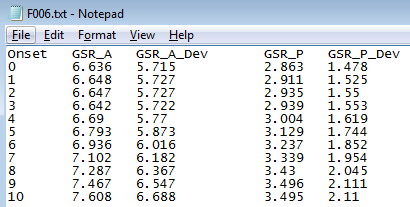
Note that the lag values and corresponding correlation values for the last run participant (DA028 in this example) are pasted at the top of the spreadsheet. These values change as the macro runs through each participant and pastes the values here before extracting the largest one and pasting it below.

## 7.3) Windowed Correlation Time Series

Calculates a series of correlation coefficients within windows of a user-specified length. For example, a 20-second window begins by calculating the correlation for 0-20 seconds, then shifts to 1-21 seconds, then 2-22 seconds, etc.

For this macro, the output is a series of text files containing a “time series” of correlations, where each correlation value corresponds to a window rather than a single onset value. This “time series” is generated for each participant. These files are saved to a new subfolder within the input folder pathway.

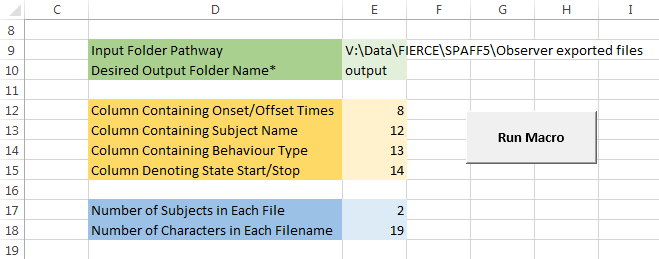
**Input:**  


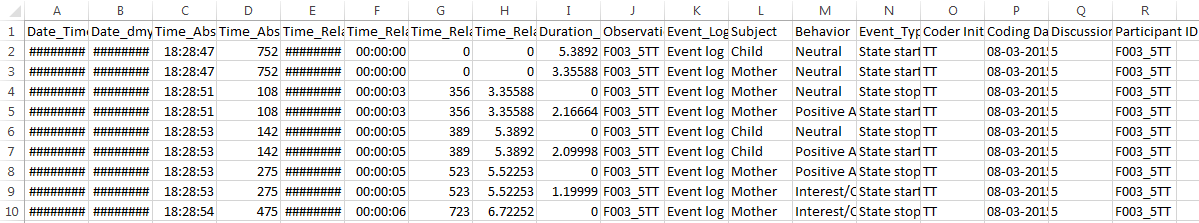
**Example of One Original File: Resulting File:**

# 8) Noldus Observer Event Logs to TRJ Conversion

Cycles through the Noldus Observer XT event logs of all participants and creates a text file to indicate the respective behaviours of a mother-daughter dyad (for example) at a given time point. Resulting file contains 3 new columns: onset, mother behaviour, and daughter behaviour. The input folders for this macro are the Excel “Event Log” spreadsheets exported from Noldus Observer XT 10.5.

**Note:** Time column contains only times at which behaviours were observed, not a complete time series.

**Input:**

**Example of One Original File (Event Log):**

**Resulting File:**