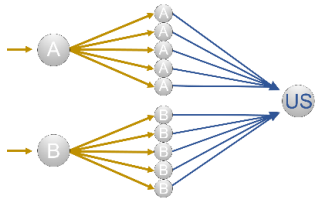
ALTSIM

MANUAL & RELEASE NOTES



ALTSim is a MATLAB-based simulator of several associative learning models. It provides graphical user interfaces for specifying all relevant parameters as well as the exact stimulus sequences.

ALTSim, the Associative Learning Theories Simulator, was initially intended for our own internal use in the associative learning lab of the Philipps-University Marburg. The main aim has always been to provide undergraduate and post-grad students a tool to discover modern and complex associative learning models without the need to know anything about programming. As many of our research questions concern the comparison of predictions of these models, a second aim has been to simplify generating comparable predictions by the models of interest to our research. These aims explain for example why large amount of comments in the code are in German and why we only included trial-based models (but no real-time model).

By version 3, we thought that ALTSim might be useful for other researchers as well. We therefore uploaded it to our servers and published a paper in Behaviour Research Methods, 41(1), 29-34, doi:10.3758/BRM.41.1.29. This paper still correctly describes the main structure of the simulator as well as the variables and the parameters.

Since then we have added additional features and changes to the layout of the user interfaces, so now it seemed appropriate to refer to the new version as 4.0 and to provide a complete manual instead of only a list of changes. The first part of the following describes how to use ALTSIM 4.0. The release notes for previous versions follow below.

More information on the models and the specific implementation can be found in the following papers:

- Thorwart, A. & Lachnit, H. (2019). Inhibited Elements Model — Implementation of an associative learning theory. *Journal of Mathematical Psychology*, 102310, https://doi.org/10.1016/j.jmp.2019.102310.
- Thorwart, A., Schultheis, H., König, S. & Lachnit, H. (2009). ALTSim: A MATLAB simulator for current associative learning theories. *Behavior Research Methods*, 41(1), 29-34,doi:10.3758/BRM.41.1.29
- Schultheis, H., Thorwart, A., & Lachnit, H. (2008a). HMS: A MATLAB simulator of the Harris model of associative learning. *Behavior Research Methods*, 40, 442-449.doi:10.3758/BRM.40.2.442, simulator
- Schultheis, H., Thorwart, A., & Lachnit, H. (2008b). Rapid-REM: A MATLAB simulator of the replaced elements model. *Behavior Research Methods*, 40, 435-441. doi:10.3758/BRM.40.2.442, Simulator

Anna Thorwart, January 2020

anna.thorwart@staff.uni-marburg.de

MANUAL

We will use the basic nomenclature and concepts of Pavlovian conditioning paradigms and associative learning models (e.g., lambda of the US).

HOW TO INSTALL AND START ALTSIM

We provide ALTSim as original Matlab code, as Matlab app as well as compiled stand-alone versions for Windows, Linux, and Mac. As we programmed ALTSim on a Windows PC, please let us know if something is not working on Linux or Mac.

THE MATLAB APP

- *To install*, run "ALTSim app.mlappinstall" in Matlab or go the "App" Tab in Matlab, click "Install App" and select the "ALTSim.mlappinstall".
- *To start*, click on the button in your Apps-tab.
- *The advantage*: You can use all features provided by Matlab, such as the figure editor. You can start the app independently from the current folder in Matlab.
- *The disadvantage*: You need a license for Matlab. For the Inhibited Elements Model you additionally need a license for the Statistics and Machine Learning Toolbox.

THE ORIGINAL MATLAB CODE

1. *To install, s*ave everything to a folder of your choice.
2. *To start*, run ALTSim.m
3. *The advantages*: You can use all features provided by Matlab, such as the figure editor. You can easily read and change the code.
4. *The disadvantages*: You need a license for Matlab. For the Inhibited Elements Model you additionally need a license for the Statistics and Machine Learning Toolbox. You need to make to the ALTSim folder the current folder in Matlab in order to run ALTSim (unless you put in on the Matlab search path).

THE STAND-ALONE VERSIONS

1. *To install*, run "MyAppInstaller_web.exe". You will need an internet connection to download the appropriate Matlab runtime during the installation process.
2. *To start*, run ALTSim.exe
3. *The advantage*: You do not need any Matlab license.
4. *The disadvantage*: You cannot see the code and cannot use the other features provided by Matlab, in particular the figure editor.

WHAT ELSE DO YOU NEED BEFORE STARTING

MANDATORY INPUT FILES

All information for and of a simulation is saved in simple text files. The files can have arbitrary names, but ALTSim assumes certain file extensions for different file types. Details on their content and structure are explained further below when we describe how to use them during simulations.

[DEFAULT].CONF

A text file that specifies *default values for most parameters*.

This is a mandatory input file. One example is included in the installation package and you should not alter the general structure of the file, but only the numbers representing the parameter values. If you cannot find it or so not have access where ALTSim saved it, you can always download it from the website: https://www.uni-marburg.de/fb04/team-lachnit/mitarbeiter/thorwart/simulators/altsim/altsim4_0/default.conf

The file can be accessed via the menu of each user interface and then edited in an editor. For this to work, you need to specify where the parameter file is saved. This has to be done at least once, normally only when you start ALTSim for the first time.

[TRIALMATRIX].MTX

A text file that contains the *trial matrix* that provides information about the name for each trial type as well as the lambda and beta of the US in this trial.

This is a mandatory input file. We provide some examples on our website.

[STIMULUSCODE].CDE

A text file that contains the *stimulus code* that provides information about the CS in each trial type.

This is a mandatory input file. We provide some examples on our website.

OTHER INPUT FILES

[TRIALSEQUENCE].SEQ

A text file that provides information about the *sequence of training trials*.

This is an optional input file as sequences can also be generated by ALTSim.

[PROPORTIONMATRX].PRO (mandatory only if you simulate the Replaced Elements Model)

A text file that contains the *proportion matrix* that contains the proportions of elements of a CS that will be replaced in the context of another CS.

We provide some templates in a folder called “pro_files” in the installation package.

STEP 1: INITIALIZE TRIALS

ALTSim is a trial-based simulator. It assumes that any learning experiment consists of repeatedly presenting different kind of CS-US combinations, each combination known as a trial type.

Thus, the first step of any simulation is to provide and generate the necessary information on the trial types and the frequency and order in which they should be presented within the simulation. After specifying the parameters as described in the following, click the “**1. Initialize trials**” button.

Note, that every time you change any of the parameters’ values, you need to click the button again. Otherwise, the information will not be updated and transferred to the models’ user interfaces.

TRIAL TYPES (TRAINING & TEST TRIALS)

We differentiate between trial types that are “training trials”, in which learning takes place and associative weights are updated, and “tests”, for which no associative weights are updated. The latter are trials intended to test CS combinations different from the trained ones without affecting the associative weights. In the first step, one provides the same information for both kind of trial types. Which trial types are training trials and which are test only is specified later via the trial sequences.

Information on the trial types is split in two parts: the trial matrix and the stimulus code. Each trial type is defined by a line in both the trial matrix and the stimulus code.

The trial matrix (saved in a file with the extension *.mtx) provides information about the name as well as the lambda and beta of the US in this trial type. The names should not contain any spaces. The lambda and beta

should be between 0 and 1. Note the lambda and beta parameters are also specified for the "test trials" even though this information is never used.

The stimulus code (saved in a file with the extension *.cde) provides the information on the present CS in each trial type. Each CS in the experiment is represented by a column in the stimulus code and its presence should be indicated by 1, its absence with 0. Do not provide information on unique or configural cues or on elements here, as the full stimulus representation is automatically computed by ALTSim based on the different theories' assumptions.

SPECIFY FILE WITH TRIAL MATRIX AND STIMULUS CODE

- Before starting any simulation, you need to create some files because their location has to be specified in the two fields in the top panel of the main user interface.

INSPECT AND/OR EDIT TRIAL MATRIX AND STIMULUS CODE

- To inspect or edit both, click on "*Trial Viewer*". As the Matlab editor does not allow adding or removing lines and thus trial types, you have the option to *open the files* in an external editor. After changing and saving the files in the external editor, you will need to *reload the files* within the Trial Viewer.

TRAINING TRIAL SEQUENCES

A training trial sequence is the successive presentation of trials of different trial types. The second panel of the main user interface has two modes: you can either "generate" trial sequences or "load" them from a trial sequence file (*.seq).

GENERATE

A trial sequence is generated by replicating a block, which is the repeated presentation of a group of trial types. If the group of trial types changes, this defines a new phase.

SPECIFY THE FOLLOWING PARAMETER:

- *Trials trained in each block*: Specify here which of the trial types specified in the trial matrix and the stimulus code should be trained how often and at which time during the simulation. Indicate the trial type by its line number in the trial matrix and stimulus code. To manipulate the relative frequency of one trial type within one block, it can be specified several times. If you would like to have different phases during the training, in which different trial types are trained, you create a different block of trial types for each phase, separated by a semicolon. In the example in the user interface (e.g.: 1 2;3 3 2 1), trial types 1 and 2 are trained in the first part of the trial sequence equally often; later on, trial type 3 is added and presented twice as often then trial type 1 or 2. Importantly, you only need to specify training trials and not mere *test trials* as the predictions for all trial types specified in the trial matrix and the stimulus code are calculated automatically for each trial.
- *Numbers of blocks per phase*: Specify here how often a block as defined in the above field is repeated in each phase. Again, separate phases by a semicolon. In the example in the user interface (e.g.: 3;5), the first phase consists of three blocks of one trial of trials types 1 and 2; the second phase then consists of five blocks of two trials of trial type 3 and one trial of trials types 1 and 2 each.
- *Randomize trials*: The specific sequence of trials often has a strong impact on the predictions of the models. To avoid this, the sequence of trials can be (pseudo)randomized. Randomization in ALTSim is done per phase.
- *Max. number of subsequent trials with the same lambda*: If you randomize, you can specify how many subsequent trials maximally should have the same US. If randomization takes very long, you should abort and check if this restriction makes finding a valid sequence randomly very unlikely or even impossible.
- *Number of trial sequences*: This becomes important when you choose to randomize the trials. The specific sequence of trials often has a strong impact on the predictions of the models. To avoid this, several different trial sequences can be run in a simulation and the results are then the averaged predictions across all trial sequences (or subject).
- *File to save trial sequences, optional*: if you would like to save the generated sequence for later use, specify a file (with the extension *.seq) here.

LOAD

Trial sequences can also be read in from a file (with the extension *.seq) generated either in previous simulation or by other programs.

Each line of the trial sequence file specifies the trial type trained in a trial, indicated by the trial type's line number in the trial matrix and stimulus code. An additional comma-separated column of the same length specifies the phase of the current trial. If more than one trial sequence is defined, these should be added as additional comma-separated columns of the same length before the phase column.

SPECIFY THE FOLLOWING PARAMETERS

- *Trials trained in each block*: Specify here which of the trial types defined previously should be trained how often and at which time during the simulation. This is not strictly necessary but the information is used to label the resulting figures correctly.
- *Number of trial sequences*: Specify how many sequences are defined in the file.
- *File with trial sequences:* Specify the file path to the file

STEP 2: CHOOSE THEORY

Select the theory that you would like to simulate and click the "**2. Choose theory**" button. This will transfer the trial information and the current default parameter values to the theory-specific user-interface. You can have several theory-specific user interfaces open at the same time. However, every time you change any of the trial parameters' values, you will need to click the "**2. Choose theory**" button again, too. Otherwise, the information will not be transferred to the simulation. The same is true if you change the default parameters via the menu of any of the user interfaces.

STEP 3: SIMULATE

Before starting the simulation by clicking the "**3. Simulate**" button of any theory-specific user interface, you need to specify some theory-specific parameters.

Some of these exist in all theories (like the CS's associability alpha) but often need to have different values for the different theories. Other parameters only exist in one model. In addition, you can indicate how the results of the simulation should be saved and presented.

REMAINING PARAMETER

For most parameters, you can define default values that are used if the field is empty. This is in particular convenient for the alpha parameters as their number can increase rapidly with the number of theoretical representational elements assumed by the different theories. Then, ALTSim will use the default value for all units.

(EXTENDED) CONFIGURAL MODEL OF PEARCE

ALTSim simulates Pearce (1994)'s model and the extension by Kinder & Lachnit (2003). ALTSim creates a configural unit for each trial type. If exactly the same configuration is used in several different trial types (A+, A-), there is one unit for each trial type within the simulation. As all configural units are completely activated in the appropriate trials, there is perfect "generalisation" of their already learned associations. However, only the associative weight of the presented trial type unit is changed.

SPECIFY THE FOLLOWING PARAMETERS

- *File specifying associative weights at the beginning of the simulation*: If you would like to start with all associative weights set to something else than zero at the start of the simulation, you can provide here the link to a result file of a previous simulation that contains the simulated associative weights ("*_weights.txt"). ALTSim will then read in the last line of this file and use it as the starting values for the current simulation for all sequences. For this to work, the stimulus codes of both simulation need to have the same number of columns and respective columns have to specify the same CS. This option was especially useful before we added the option to run simulations with several phases.
- *Alphas for input units:* Specify here the associability of each CS, separating each value by a space. The values are used to weight the activation of each CS's input unit and thus its contribution to the activation of output

units by the presented CS. By this, it also influences the similarity and generalisation between stimulus configurations. Note that the value needs to be greater than zero.

- *Alphas for the configural units:* Specify here the associability of the CS configuration of each trial type, separating each value by a space. This value is implemented as a learning parameter and regulates the change in associative weights for its configural unit.
- *Discrimination parameter:* Specify here the general amount of generalisation and discrimination. A value of 2 corresponds to the original Pearce (1994) model, higher values increase discriminability, lower values decrease it and increase generalisation.

RESCORLA-WAGNER WITH MODIFICATIONS

ALTSim simulates the classical model of Rescorla & Wagner (1972) as well as two extensions: unique cues or configural cues as added elements for each possible combination of the single CS (e.g., Rescorla, 1973; Whitlow & Wagner, 1972) and reduced saliences depending on the number of CS present (Redhead & Pearce, 1995).

SPECIFY THE FOLLOWING PARAMETERS

- *File specifying associative weights at the beginning of the simulation*: If you would like to start with all associative weights set to something else than zero at the start of the simulation, you can provide here the link to a result file of a previous simulation that contains the simulated associative weights ("*_weights.txt"). ALTSim will then read in the last line of this file and use it as the starting values for the current simulation for all sequences. For this to work, the stimulus codes of both simulations need to have the same number of columns and respective columns have to specify the same CS. This option was especially useful before we added the option to run simulations with several phases.
- *Alphas for CS and unique cues:* Specify here the associability or salience of each CS and unique cue, separating each value by a space.
- *Compute unique cues*: if checked, unique cues are assumed for all possible combinations of the CS component. They are created by permutation of all combinations, beginning with the two-component CS compounds containing the first component, then those containing the second component, and so on. Afterward, all combinations with three components are generated, beginning with combinations containing the first two components, then the first and the third component, and so on. All unique cues whose CS are present are activated, that is if a three CS compound ABC is presented, the unique cues AB, AC, BC, and ABC are activated as well.
- *Used modified salience equation of Redhead and Pearce (1994)*: if checked, the salience of all CS (and unique cues) is reduced in proportion to the number of present CS (and unique cues). For further details, see Equation 2 of Redhead and Pearce (1994).

REPLACED ELEMENTS MODEL

The replaced elements model of Wagner (2003) is an elemental model that assumes a certain proportion of context-dependent elements in each CS representation. Some context-dependent elements are only activated if another CS is present as well, some only if the other CS is absent: the latter are thereby replaced by the former. The proportion of these context-dependent elements is indicated in the proportion matrix.

The proportion matrix contains a line for each CS. Each line contains the proportions of elements to be replaced for this CS in the context of the other CS indicated by the column number. For example, the third number in the second line contains the proportion of replaced elements in CS 2 by CS 4. Since the numbers in each line represent proportions of to-be-replaced elements, they should lie within the closed interval [0,1].

SPECIFY THE FOLLOWING PARAMETERS

- *File specifying associative weights at the beginning of the simulation*: If you would like to start with all associative weights set to something else than zero at the start of the simulation, you can provide here the link to a result file of a previous simulation that contains the simulated associative weights ("*_weights.txt"). ALTSim will then read in the last line of this file and use it as the starting values for the current simulation for all sequences. For this to work, the stimulus codes of both simulations need to have the same number of columns and respective columns have to specify the same CS. This option was especially useful before we added the option to run simulations with several phases.
- *Alphas for CSs:* Specify here the associability or salience of each CS, separating each value by a space.

- *File specifying proportion of replaced elements:* Specify the path of the file that contains the proportion matrix. It should be a simple, space-separated text file with the extension ".pro". We provide several examples and templates.
- *Learning rule:* Specify whether the replacement should be implemented as suggested by Schultheis et al. (2008b) or by Glautier (2007).

INSPECT AND/OR EDIT PROPORTION MATRIX

- To inspect or edit the proportion matrix, click on "*view*". As the Matlab editor does not allow for adding or removing lines and thus trial types, you have the option to *open the file* in an external editor. After changing and saving the file in the external editor, you will need to "*reload the file".*

ELEMENTAL MODEL OF HARRIS

The elemental model of Harris assumes a complex stimulus representation. As ALTSim uses random numbers to determine the initial activations of the elements of each CS and the existence of connections among CSs and USs, different simulations of the model give slightly different results. For this reason, ALTSim does not provide the associative weights for the Harris Model. Instead and to get an impression of how the model will behave on average, the model is simulated several times, and the average US activation in these runs is taken as the model output. If there is more than one sequence to simulate, all sequences will be simulated within a run and then averaged.

SPECIFY THE FOLLOWING PARAMETERS

- *number of runs:* Specify how many times the simulation should be computed.
- *connection density:* Specify the proportion of connections existing between pairs of elements. Valid entries for this parameter are all real numbers from the closed interval [0,1], where 0 results in no connections at all and 1 results in complete connection of all elements.
- *attention buffer gain:* Specify the boost each element's activation receives when gaining entry into attention buffer.
- *alphas for CSs:* Specify here the associability or salience of each CS, separating each value by a space.
- *attention buffer capacity:* The capacity can be calculated automatically so that each CS will gain complete entry into the buffer when presented alone. Alternatively, you can specify a capacity as any real value above 0.
- *fraction of common elements:* Specify here the overlap of elements representing different single CSs
- *simplified learning:* Specify whether the simulation should employ the simple delta rule or the more complex learning using cutoffs (for details, see Schultheis et al., 2008a)

INHIBITED ELEMENTS MODEL

The inhibited elements model is an elemental model that assumes a certain proportion of context-dependent elements in each CS. Some context-dependent elements are only activated if another CS is absent: the latter otherwise inhibits the elements of the former so that the overall number of active representational elements is constant, or "normalized" (Thorwart et al, 2018).

SPECIFY THE FOLLOWING PARAMETERS

- *File specifying associative weights at the beginning of the simulation:* If you would like to start with all associative weights set to something else than zero at the start of the simulation, you can provide here the link to a result file of a previous simulation that contains the simulated associative weights ("*_weights.txt"). ALTSim will then read in the last line of this file and use it as the starting values for the current simulation for all sequences. For this to work, the stimulus codes of both simulations need to have the same number of columns and respective columns have to specify the same CS. This option was especially useful before we added the option to run simulations with several phases.
- *Alphas for CS*: Specify here the associability or salience of each CS, separating each value by a space.
- *Amount of normalisation*: Specify here the general amount of normalisation. The number should lie within the closed interval [0,1]: if set to zero, no elements will be inhibited and the models corresponds to the

Rescorla-Wagner model. If set to one, the proportion of inhibited elements is calculated so that the overall number of active elements stays constant across all different trial types and the number of presented CSs.

RESULTS

Results of the simulations are available both as text files and plotted in figures. Three different kinds of simulation results are available:

- *Associative* weights: associative weights of all CS, unique cues, configural etc. of the experiment at the beginning of each trial of the experiment plus at the end of the last trial.
- *Test activation:* US activation by the CSs of any trial type at the beginning of each trial. This calculates the (hypothetical) activation of the US by the CSs in any trial types, test and training, defined in the trial matrix and/or stimulus code. For example, even if you train only AB+ trials, this will give you the predicted US activation for any other CS combination, for example A? or ABC? throughout the simulation. Thus, it allows training and automatically testing for US activation at the same time.
- *Trained activation:* US activation only by CSs occurring in current trial, which is proportional to what is measured as CR in each trial in an experiment.

SPECIFY THE LOCATION AND TO SAVE RESULTS FILES

If you specify a location and name, three separate files will be generated, containing the three different kinds of simulation results. All files are simple space separated text files.

- *Associative* weights: the extension "_weights.txt" is added to the specified file name. There is a column for each theoretical element assumed by the different theories. For the specific order of the elements, please check out the appropriate papers. Each line in the file contains the associative weights at the beginning of a trial. The file has one line more than trial in the simulations as the associative weights after the last trial are calculated as well. Note that we do not provide this for the Harris Model as the introduction of randomness into the stimulus presentation renders the specific associative weights meaningless.
- *Test activation*: the extension "_testactivation.txt" is added to the specified file name. There is a column for each trial type of the simulation. An additional column indicates the phase of the current trial. Each line in the file contains the results for one trial of the entire simulation, for example, the second line contains the activation of the US unit by any CS configuration after one trial of training.
- *Trained activation:* the extension "_trainedactivation.txt" is added to the specified file name. There is a column for each trial type of the simulation. An additional column indicates the phase of the current trial. Each line in the file refers to the trial number of each trial type in one phase, for example, the second line contains the activation of the US unit by CSs of a certain trial type the second time this trial type was trained. If the trial type was not trained in this phase at all or if there are less trials of this trial type in this phase then the overall number of trials, there is no result for this trial type in this line and NaN for "Not a Number" is written instead.

SPECIFY WHICH FIGURES TO SHOW

Figures are standard Matlab figures that can be edited and inspect with the Matlab Plot Tools. The later are not available in the compiled versions of ALTSim.

- *Plot associative weights at the beginning of each trial*: Creates a figure with a line for each theoretical element assumed by the different theories. For the Pearce Model, the associative weights for identical configural units are already summed up. For the specific order of the elements in the other models, please check out the appropriate papers. We do not provide this for the Harris Model as the introduction of randomness into the stimulus presentation renders the specific associative weights meaningless.
- *Plot US activation by CSs of any trial type in each trial:* Creates a figure with the test activation. Stages are plotted in separate axes. In each, the number on the x-axis refers to the trial number across the entire phase.
- *Plot US activation by CSs of currently trained trial:* Creates a figure with the trained activation. Stages are plotted in separate axes. In each, the number on the x-axis refers to the number of training trials for each trial type in each phase. (Thus, if trial types are differently often trained in a phase, the line will be differently long.)

.

RELEASE NOTES

ALTSIM 3.4

ALTSim 3.4 is the first version we put out there and is therefore the one described in:

Thorwart, A., Schultheis, H., König, S. & Lachnit, H. (2009). ALTSim: A MATLAB simulator for current associative learning theories. *Behavior Research Methods*, 41(1), 29-34, doi:10.3758/BRM.41.1.29

ALTSIM 3.7

CHANGES:

1. The previous versions of the Pearce Model contained an error: If exactly the same stimulus was used in several different trial types (e.g., A+, A-, A?), learning was inappropriately accelerated. This has been correct.

NEW FEATURES

1. There is now the possibility to enter additional alphas for Pearce's input unit following Equation (9) of Pearce (1994).
2. In the REM, instead of the general alpha parameter, it is now possible to enter an alpha for each component. We assume that a highly salient component will activate all its elements (context independent and dependent) more strongly than a less salient component.

ALTSIM 3.9

CHANGES

ALTSIM 3.9 WAS BUILT WITH MATLAB 2015B

There have been some major changes to Matlab's graphics system in the last year, which means that backward compatibility has become an issue. ALTSim 3.9 was programmed in Matlab 2015b and it seems that there are already problems when Matlab 2014b is used. We are trying to address this.

In the meantime, if you encounter problems, we recommend using the compiled stand-alone version as this does not rely on any Matlab installation.

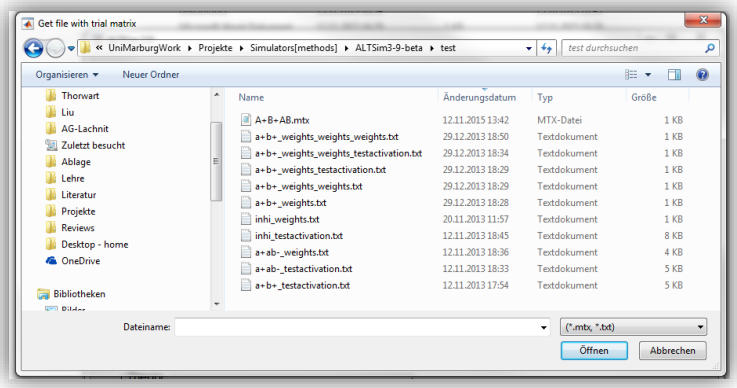
ALTSIM IS NOW A MATLAB APP

The non-compiled version of ALTSim is now available as a Matlab App. To install it, run "ALTSim.mlappinstall" in Matlab 2015b or go the "App" Tab in Matlab and click "Install App".

CHANGES TO FILE TYPES

In order to make it easier to distinguish between and organise different files, we introduced standard name extensions for matrix files (*.mtx) and for the stimulus code file (*.cde). The files should still be saved and formatted as simple text files that were created by any text editor.

You can still use .txt as extension and/or your existent files as files with the .txt – file extension will be shown by default too.

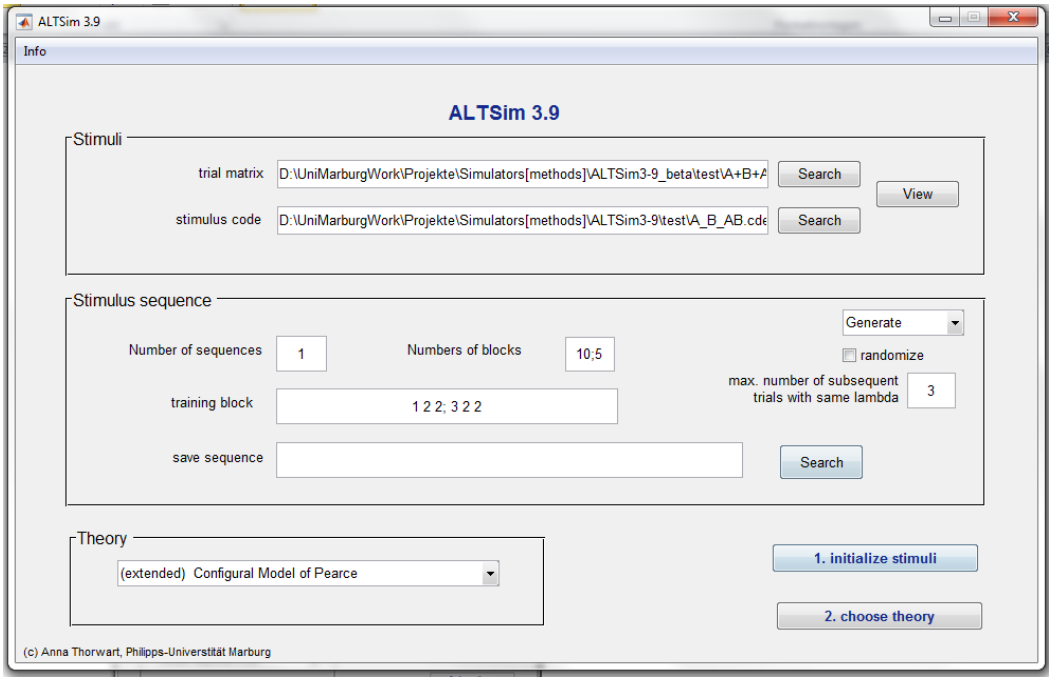


NEW FEATURES

EXPERIMENTS WITH SEVERAL DIFFERENT PHASES

It is now easier to simulate experiments with several different stages. Just separate the description of the different stages with a semicolon in the “Number of blocks” and the “training block” field.

In the example, the experiment consists of 2 phases. In the first phase, 10 blocks of Trial Type 1 and of twice Trial Type 2 are trained; in the second phase 5 blocks of Trial Type 3 and again of twice Trial Type 2 are trained. The



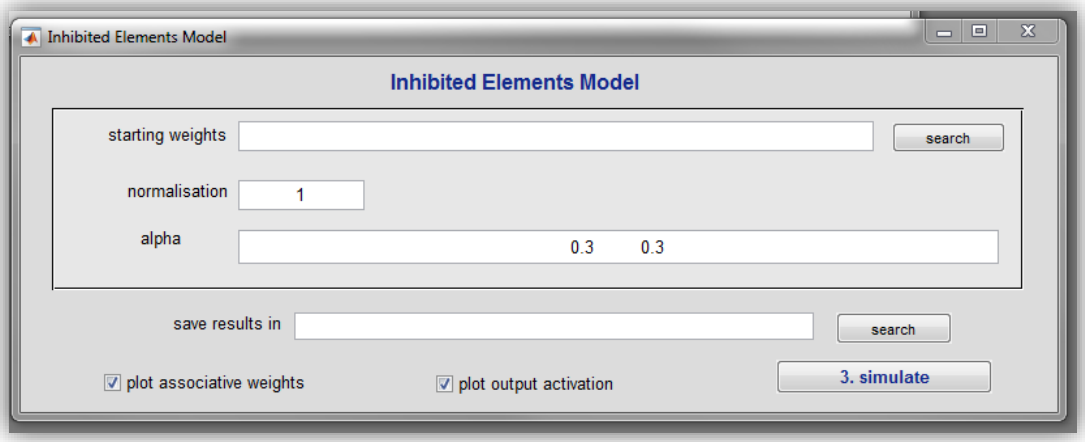
trial sequences of each phase are generated independently from each other and the restrictions on the max number of subsequent trials with the same lambda will be applied to all phases equally.

There is also now the possibility to abort the randomisation in case it takes too long, for example if your restriction is too strict.

The results of the complete training will be shown in one figure and saved in one file.

NEW MODELS

ALTSim 3.9 includes an "Inhibited Elements Model" based on Wagner and Brandon (2001, also see Brandon, Vogel & Wagner, 2000). In contrast, to Wagner and Brandon' s model, it includes a parameter that controls the



amount of normalisation and can have a value between 0 and 1.

When normalisation=1, the summed activation of all elements representing all present CS is always 1, i.e. the activation of each element decreases the more CS are present (complete normalisation). When normalisation=0, each CS is presented by one element that is always fully activated to 1, i.e. there is no normalisation and the representation is completely context-independent.

For details on the implementation, please contact me, anna.thorwart@staff.uni-marburg.de. (There also will be a published paper about it at some point.)

ALTSIM 3.10

CHANGES

RESOLVED COMPATIBILITY ISSUES

We re-builded the whole simulator ALTSIM 3.9 using older versions of Matlab (mainly 2013a) to resolve the compatibility issues. This version was then tested with 2013a and 2016a.

We would suggest not using it in 2015b (without SP1) as this release caused the problems in ALTSim3.9

MINOR CHANGES

- We fixed some bugs in the randomisation and saving functions.
- We changed the layout of the main interface.

ALTSIM 4.0

- We changed the layout of the main interface.
- Information about phase added to output files

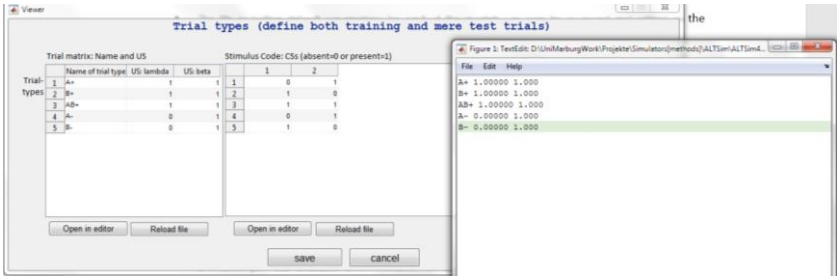
NEW FEATURES

EASIER TO CHANGE DEFAULT PARAMETERS

- The file including default parameters for each of the models can now be accessed and edited via the menus of each user interface and then edited in the editor. For this to work, you need to specify once where the parameter file is saved.

NEW STIMULUS VIEWER

- We changed the layout of the Stimulus Viewer and added the option to open the files for the trial matrix and the stimulus code in an external editor. This allows adding and removing entire rows or columns of the tables. After changing the files in the external editor, you will need to reload the files within the Stimulus Viewer. (Please let us know if this is not working.)

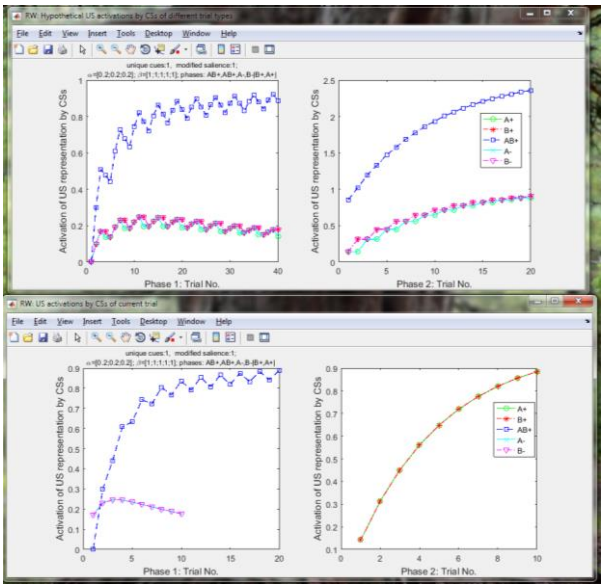


-

ADDITIONAL GRAPHIC OF OUTPUT ACTIVATION

- Previously, only one type of activation plot was available that showed the hypothetical activation of the US by each CS configuration in any trial types. We added a new type of plot that shows the US activation only by CSs occurring in current trial, which is what can be measured as CR in the empirical experiment. The x-axis refers then to the trial number of each trial type and not the entire experiment.

- Stages are plotted in two separate axes for both activation plots.



CHANGES

MINOR CHANGES

- We changed the layout of most user interfaces.

- The sequence file has now an additional column indicating the phase number.
- The hypothetical US activations by CSs of different trial types is now saved in the file ending with '_testactivation.txt'. The number in the last column of this file indicates the current phase number.
- The US activation by CSs of current trial is saved in the file ending with '_trainedactivation.txt'. The number in the last column of this file indicates the current phase number.

REFERENCES

Brandon, S., Vogel, E., & Wagner, A. (2003). Stimulus representation in SOP: I. Theoretical rationalization and some implications. *Behavioural Processes*, 62(1–3), 5–25.

Glautier, S. (2007). Simulation of associative learning with the replaced elements model. *Behavior Research Methods*, 39, 993-1000. doi:10.3758/bf03192995.

Harris, J. A. (2006). Elemental representations of stimuli in associative learning. *Psychological Review*, 113, 584-605. doi:10.1037/0033-295X.113.3.584.

Kinder, A. & Lachnit, H. (2003). Similarity and discrimination in human Pavlovian conditioning. *Psychophysiology*, 40, 226-234. doi:10.1111/1469-8986.00024.

Pearce, J. M. (1994). Similarity and discrimination: A selective review and a connectionist model. *Psychological Review*, 101, 587-607. doi:10.1037//0033-295x.101.4.587.

Redhead, E. S. & Pearce, J. M. (1995). Similarity and discrimination learning. *Quarterly Journal of Experimental Psychology*, 48B, 46–-66.

Rescorla, R. A. (1973). Evidence for "unique stimulus" account of configural conditioning. *Journal of Comparative and Physiological Psychology*, *85*(2), 331–338. doi:10.1037/h0035046

Rescorla, R. A., & Wagner, A. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and non-reinforcement. In A. H. Black & W. F. Prokasy, A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning. II. Current research and theory* (pp. 64–99). New York: Appleton-Century-Crofts.

Schultheis, H., Thorwart, A., & Lachnit, H. (2008b). Rapid-REM: A MATLAB simulator of the replaced elements model. *Behavior Research Methods*, 40, 435-441. doi:10.3758/BRM.40.2.442, Simulator

Schultheis, H., Thorwart, A., & Lachnit., H. (2008a). HMS: A MATLAB simulator of the Harris model of associative learning. *Behavior Research Methods*, 40, 442-449.doi:10.3758/BRM.40.2.442, simulator

Thorwart, A., & Lachnit, H. An Inhibited Elements Model – computational implementation and simulations of an associative learning theory, *in preparation*.

Thorwart, A., Schultheis, H., König, S. & Lachnit, H. (2009). ALTSim: A MATLAB simulator for current associative learning theories. *Behavior Research Methods*, 41(1), 29-34,doi:10.3758/BRM.41.1.29

Wagner, A. R. (2003). Context-sensitive elemental theory. *Quarterly Journal of Experimental Psychology*, 56B, 7-29. doi:10.1080/02724990244000133.

Wagner, A. R., & Brandon, S. E. (2001). A componential theory of Pavlovian conditioning. In R. R. Mowrer & S. B. Klein, R. R. Mowrer & S. B. Klein (Eds.), Handbook *of Contemporary Learning Theories* (pp. 23–63). Mahwah,NJ: Erlbaum.

Whitlow, J. & Wagner, A. (1972). Negative patterning in classical conditioning: Summation of response tendencies to isolable and configural components. *Psychonomic Science*, 27, 299-301. doi:10.3758/bf03328970.