**README Effort Allocation Task**

Table of Contents

[Introduction 2](#_Toc27577610)

[Script description EAT\_main.m 2](#_Toc27577611)

[Part 1: Settings 3](#_Toc27577612)

[Part 2: Timings 5](#_Toc27577613)

[WOF specific timings 5](#_Toc27577614)

[Overview other code sections 6](#_Toc27577615)

[Part 3: Input from console 6](#_Toc27577616)

[Part 4: fMRI settings 6](#_Toc27577617)

[Part 5: load required files 6](#_Toc27577618)

[Part 6: Psychtoolbox and screens 6](#_Toc27577619)

[Part 7: general image settings 6](#_Toc27577620)

[Part 8: WoF specific settings 7](#_Toc27577621)

[Part 9: VAS specific settings 7](#_Toc27577622)

[Part 10: Training 7](#_Toc27577623)

[Part 11: Input device unrelated variables 7](#_Toc27577624)

[Part 12: Input device unrelated variables 7](#_Toc27577625)

[Part 13: Input device unrelated variables 7](#_Toc27577626)

[Part 14: Text instructions 7](#_Toc27577627)

[Part 15: start fMRI procedure 7](#_Toc27577628)

[Part 16: actual trials 7](#_Toc27577629)

[Part 17: end of trial 8](#_Toc27577630)

[Part 18: save data 8](#_Toc27577631)

[Output description 9](#_Toc27577632)

[Supporting files 9](#_Toc27577633)

[Condition files 9](#_Toc27577634)

[Folder structure 10](#_Toc27577635)

[Instructions 11](#_Toc27577636)

[Jitters 11](#_Toc27577637)

[Training 11](#_Toc27577638)

[VAS/PANAS 11](#_Toc27577639)

[WoF 11](#_Toc27577640)

[References 11](#_Toc27577641)

Current version: 2.0 “EAT at the Zoo”[[1]](#footnote-1)

# **Introduction**

The Effort Allocation Task (EAT) is a cost-benefit paradigm that allows participants to work for food and monetary reward though effort in the form of button presses or grip force. The output from this task allows for the estimation of effort vigor and maintenance. It is a task that can be used for investigating motivational processes. For more details on the first version of the task (frequency only) see the preprint by Neuser et al. (2019): **doi:** <https://doi.org/10.1101/789982> . The task is based on an original task by Meyniel et al. (2013).

The current version not only allows for grip force, it also enables the following major features:

*Uncertainty*

Uncertainty is interesting for the following reason: “…trials with uncertainty allow insight into the role of the expected reward value. When uncertainty exists regarding the amount of effort that the participant needs to exert to obtain the reward, the best strategy is to exert the amount of effort they are willing to spend for the reward that is offered in that trial. Therefore, the effort exerted reflects the expected reward value during a trial, and changes in the amount of effort a participant is exerting reflect changes in the expected reward value over time.” (Quote lab rotation report development of grip force EAT).

*fMRI Compatibility*

The original, frequency-based version had an input device (X-Box 360 controller) that is not MR-compatible. The grip force version solves this problem.

*Wheel of Fortune (WoF)*

An addition to the paradigm, that aims to induce different moods for different trials. This could help uncover the influence of mood induction on effort (maintenance and vigor).

# **Script description EAT\_main.m**

In principle, there are two parts in which settings need to be adapted to customize the EAT: the settings section and the timing section (Part 1 and Part 2). There are some other points that need to be taken into consideration, that are not part of these sections:

* Participant IDs are filled up with 0 at the front of the number, until the ID consists of a total of 6 numbers. This operation is done in *Part 3: input from console.*
* The variables specifying whether the text should be mirrored horizontally or vertically are called ‘flp\_flg\_hrz’ and ‘flp\_flg\_vrt’ and can be set to 1 to mirror the text for MR purposes. This operation is done in *Part 4: fMRI settings*
* All supporting files and images are loaded into PTB in *Part 5: load required files* and should be adapted if files with different naming schemes are to be loaded (e.g. other images should be loaded).
* Text instructions that contain variable values (e.g. personalized food reward through the bidding phase) are, for a large part, not in the language file, and need to be changed within *Part 14: text instructions.* Similarly, other hard coded texts are present in section 17 and at the beginning of section 16 (16.01: breaks).

## *Part 1: Settings*

This part can be either set in the EAT\_main.m code, or with the Create\_SettingStructs.m. The code is identical, but the latter has the advantage that you can save the settings once, and then load them into EAT\_main directly. This could be useful to ensure consistency. Be sure to update the name of the .mat file with your settings in EAT\_main.m, so the correct settings are loaded.

The following will detail all settings that are found at the beginning of the EAT in part 1, and that can be used to customize the EAT script to ones needs. Settings are elaborated on in order of appearance in the script.

|  |  |
| --- | --- |
| settings.do\_fullscreen = 1; | 0 gives a small screen useful for debugging or testing. 1 Gives a full screen, which is automatically the second monitor (if one is connected) |
| settings.do\_fmri = 1; | Includes MR trigger and timing recordings if set to 1. |
| settings.debug = 0; | Theoretically allows for debugging without attaching an input device if set to 1. Still needs to be fully tested. |
| settings.do\_gamepad = 1; | Uses gamepad (Xbox controller) as input device if set to 1. In that case, the effort exerted is in the form of button press frequency. Uses grip force device as input device if set to 0, in which case grip force is the type of effort exerted. |
| settings.do\_WOF = 1; | Includes Wheel of fortune if 1. |
| settings.do\_VAS = 1; | Includes questions in form of VAS at the end of each trial if 1 (specifically set which questions should be included later in the script). |
| settings.do\_val\_cal = 1; | Includes calibration of relative reward value, so that the exchange rate of points to cents/kcal reflects the difference between the value of money and food for the participant given their current satiety level. For example. If a participant is willing to work harder for money then food, and if we want them to work equally hard on food trials, we should offer them more kcal than cents for the same effort. This setting adds 6 small bidding phases after the training. Based on these trials, a ‘fair’ exchange rate is computed. |
| settings.do\_timelimit = 1; | Includes a time limit for the VAS questions if 1. Waits for input from mouse or controller before moving to the next section of 0. |
| settings.lang\_de = 1; | If multiple language versions are available, 1 sets the language of the instructions to German, 0 sets the language of the instructions to English (English language files pending) |
| settings.do\_feedback = 1; | Includes feedback after each trial, showing how many points the participant has won in that specific trial. Used mainly to indicate win if uncertainty is used (see “Supporting files for instructions on how to introduce uncertainty into the experiment”) |
| settings.train\_trials = 3; | Amount of training trials used to estimate the maximum/minimum effort. 2 trials is mainly used for frequency based EAT, 3 for grip force based EAT, although either 2 or 3 will work in both versions. |
| subj.study = 'TUE003'; | (Short) indicator of project. Is used to load/save files with appropriate project names. |
| subj.runID = '1'; | The EAT is always a single run. Run here means the amount of times the entire task is run in a single session (on a single day). This is required for our naming format. |
| settings.VAS\_input = 0; | Determines the input for the VAS. 0 is for mouse input (compatible with GFD), 1 is for joystick input (compatible with X-Box controller) |
|  |  |
| settings.VAS.exhaustion = 1; | Turns on the VAS ‘exhaustion’ question at the end of each trial (1) or leaves it out (0). |
| settings.VAS.wanting = 1; | Turns on the VAS ‘wanting’ question at the end of each trial (1) or leaves it out (0). |
| settings.VAS.happy1 = 1; | Turns on the VAS ‘happy’ question at the end of each trial (**before** feedback!) (1) or leaves it out (0). |
| settings.VAS.happy2 = 0; | Turns on the VAS ‘happy’ question at the end of each trial (**after** feedback!) (1) or leaves it out (0). |
| settings.value\_money = 17; | Cents earned with 100 money points |
| settings.value\_food = 17; | Kcal earned with 100 food points |
| settings.clckforce = 20000; | Force only: force that needs to be exerted to click through the instructions (used inside the MR-scanner) |
| settings.lang\_de = 1 | Sets language (1 German, 0 English: **English is not implemented yet!** No English language files exist): Make sure to **load appropriate instruction file(s)** in the if … else clause at the end of Part 1.. |

## *Part 2: Timings*

The timings have the same structs for fMRI and behavioral sessions. The if… else… clause is there so that settings do not need to be changed if the code is run from the same cloud file.

|  |  |
| --- | --- |
| timings.trial\_length = 24; | Length (in seconds) of a single trial |
| timings.break\_length = 15; | Length (in seconds) of a single break |
| timings.feedback\_length = 2.5; | Length (in seconds) of a feedback block |
| timings.fix1\_length = 0; | Base length (in seconds) of the first fixation cross (between end trial and VAS/feedback screens) |
| timings.fix2\_length = 1.5; | Base length (in seconds) of the second fixation cross (between end of VAS/feedback and start new trial) |
| timings.bidding\_length = 5; | Length (in seconds) of a single bidding trial |
| timings.VAS\_rating\_duration = 3.2; | Time to react to VAS scales (in seconds) |
| timings.number\_breaks = 2; | Number of breaks during the entire paradigm |
| timings.number\_trials = 64; | Number of trials in the entire paradigm. This does not change the actual amount if trials, it is, however, required for the correct computation for the placements of the breaks |
| timings.avrg\_jttr\_ball = '1'; | The average jitter time for the presentation of the reward, before the ball appears. Needs to be a string as it is used to load the correct jitter file. |
| timings.avrg\_jttr\_fix1 ='1.5'; | The average jitter time for the presentation of the first fixation cross. Needs to be a string as it is used to load the correct jitter file. |
| timings.avrg\_jttr\_fix2 = '3'; | The average jitter time for the presentation of the second fixation cross. Needs to be a string as it is used to load the correct jitter file. |
| timings.max\_jttr\_ball = '4' | The largest value the ball jitter can take. Needs to be a string as it is used to load the correct jitter file. |
| timings.max\_jttr\_fix1 = '12'; | The largest value the first fixation cross can take. Needs to be a string as it is used to load the correct jitter file. |
| timings.max\_jttr\_fix2 = '12'; | The largest value the second fixation cross can take. Needs to be a string as it is used to load the correct jitter file. |
|  |  |

### WOF specific timings

|  |  |
| --- | --- |
| timings.nmbr\_trls\_to\_WOF | Number of times the WOF appears |
| timings.time\_to\_start | Lag between button press and the start of the spinning of the WoF |
| timings.show\_wheel | How long (secs) the wheel is shown (spinning phase) |
| timings.show\_feedback | How long (secs) the feedback after the wheel is shown (feedback phase) |

# **Overview other code sections**

## *Part 3: Input from console*

This part of the code has the following functions (in order):

1. Request subject and session information from experimenter (training vs. experiment, participant ID and session ID. Optionally: WoF randomization (a or b) for sessions run with WoF)
2. Convert this information to the appropriate formats (e.g. num2str, abbreviations etc.). Participants ID are filled with zeros at the front of the number to make a complete, 6 number ID.
3. Get operating system.
4. Cursor is hidden

## *Part 4: fMRI settings*

This part of the code specifies the triggers and other MR-specific variables. It also uses ‘flp\_flg\_hrz’ and ‘flp\_flg\_vrt’ to perform mirroring on the text for the MR scanner.

## *Part 5: load required files*

This part of the code loads all the different supporting files. These are in order:

1. Condition files
2. Jitter files. These are only loaded correctly if the avrg\_jttr and max\_jttr are correctly filled (as strings!) in in *Section 2: Timings.* See ‘supporting files’ for more information on how to create jitters that are supported by the EAT\_main.m script.
3. Images. If any other images need to be loaded, the names of the images need to be adapted in this section.

## *Part 6: Psychtoolbox and screens*

This part sets all the PTB and screen settings, such as the colors that are used in the paradigm.

## *Part 7: general image settings*

This part sets the parameters for all the different element on the screen, that apply to both grip force and frequency tasks, such as the position of text, the position of the tube, the size of the ball etc.

## *Part 8: WoF specific settings*

Pending.

## *Part 9: VAS specific settings*

Loads VAS jitters and shuffles them.

## *Part 10: Training*

If the round is a training round, this code will launch into the training script. For a description on this script, see the *Training* paragraph within the **supporting files**section.

## *Part 11: Input device unrelated variables*

Contains all the variables that are needed to run the EAT for both frequency-based and grip force-based input.

## *Part 12: Input device related variables*

**12.1:** Variables that are required when the frequency version is run, but not when the grip force version is run.

**12.2:** Variables that are required when the grip force version is run, but not when the frequency version is run.

## *Part 13: Uncertainty*

Positioning of uncertainty box.

## *Part 14: Text instructions*

Contains all the instructions that will be shown before the start of either the ‘real’ training, or the experiment. As of now, texts that have variable values in them are not loaded directly, so some of the texts need to be adapted within this section, if need be. This is something that will be updated in later versions.

## *Part 15: start fMRI procedure*

Listen for trigger and start getting MR-related timing values.

## *Part 16: actual trials*

**16.01:** starts break at the appropriate times. The text used for the breaks can be changed in this section.

**16.02:** if WoF is used: insert PANAS before first trial.

**16.03:** update all the trial settings before trial start (conditions, difficulty etc.).

**16.04:** show screen with the incentive and empty tube. Wait for seconds specified by ball jitter.

**16.05:** add difficulty bar. This feature is currently not in operation, however, there can also be a break between this and the actual trial start by adding in a WaitSecs at the end of this section.

**16.06:** Actual trial start.

**16.06.1:** Draw all static elements on display.

**16.06.2:** Drawing of ball through grip force device.

**16.06.3:** Drawing of ball through frequency values.

**16.06.4:** Fluctuating variables update: not input device specific.

**16.06.5:** Input query frequency.

**16.06.6:** Input query grip force device.

**16.07:** End of trial, compute wins and payouts.

**16.08:** Fixation cross 1.

**16.09:** VAS questions that are turned on (see settings) are displayed in the order: exhaustion, wanting, happy1. If WoF: intertrial PANAS questions are displayed in this part of the code.

**16.10:** Feedback on trial is given.

**16.11:** Final VAS question is presented if VAS is turned on (happy2)

**16.12:** Fixation cross 2.

**16.13:** Prepare all output structures. Save temporary data. For more insight into the output, see the section **output description***.*

**16.14:** Clear variables for the initiation of the new trial.

## *Part 17: end of trial*

Update the Maximum effort exerted if it is in the training phase. Prepare final feedback screen with wins. Show final feedback screen(s) with wins. Final feedback text can be adapted in this section. Screen is closed through mouse click.

## *Part 18: save data*

All data is finally saved here in the appropriate folders. Temporary backup files are deleted.

# **Output description**

The most important data structure is output.data\_mat. The variable names can be found in the struct output.variable\_labels. The meanings of the labels (and thus also the meaning of the columns) are described below:

1. Participant ID
2. Session ID
3. Trial number
4. Maximum exerted effort
5. Minimum exerted effort (always 0 in frequency, only required for grip force)
6. Time relative to trial start
7. Absolute exerted effort
8. Relative exerted effort (percentage of maximum exerted effort)
9. Difficulty
10. Reward type (0 for food, 1 for money)
11. Reward magnitude (1 or 10)
12. Uncertainty (0 for certainty of difficulty, 1 uncertainty of difficulty)
13. Number of seconds spent above difficulty threshold
14. Points won

All the VAS responses can be found in output.VAS\_per\_trial, in which the rows have the following meanings:

1. Trial number
2. Rating exhaustion (NaN if question not asked)
3. Rating wanting (NaN if question not asked)
4. Rating happy 1 (NaN if question not asked)
5. Rating happy 2 (NaN if question not asked)

# **Supporting files**

## *Condition files*

Condition files are .mat files that currently follow the following naming convention, which is as follows for training files:

**EATTrain\_cond\_(subj.study)\_(PPID)\_S(sessID)\_R1.mat**

And as follows for “real” experimental trials

**EATExp\_cond\_(subj.study)\_(PPID)\_S(sessID)\_R1.mat**

In which subj.study is the abbreviation for the study (which can be adapted in *Part 1: settings*) PPID is the participant ID that is inserted in the console, and sessID is the session ID that is inserted in the console.

Condition files need four columns for them to be useful. The row index indicates the trial. The meaning of these columns is as follows

1. Difficulty of the trial
2. Type of reward (0 for food, 1 for money)
3. Magnitude of reward (1 for low, 10 for high, corresponds to value of a second of effort)
4. Uncertainty (0 for certain, 1 for uncertain)

Even if uncertainty is not desired, this fourth column should be filled with zeros to prevent the code from crashing. Multiple example scripts that create condition files can be found in the directory. They all start with: “Create\_Cond” followed by the number of trials and whether uncertainty is included or not. E.g. **Create\_Cond\_48T\_Cert.mat** creates condition files with 48 trials, but without uncertainty. **Condition files should be made before starting the experimental phase.** The naming of new scripts that create condition files is irrelevant to the functioning of EAT\_main.m.

The existing creation files require the following input to create compatible files:

* subj.studyID: The abbreviation for the project in which the task is used (e.g. TUE002)
* subj.study\_part\_ID: The session ID for which the condition files need to be made.
* start\_range: The first participant ID for which a condition file should be made.
* id\_range: The last participant ID for which a condition file should be made.

If these values are correctly inserted, all existing scripts should make condition files that are compatible with our naming conventions.

## *Folder structure*

The EAT requires the following sub-folders to function:

* A folder named **backup** in which temporary and backup files are stored during the experiment.
* A folder named **conditions** which stores the condition files.
* A folder named **data** in which the final data files are stored.
* A folder named **jitters** which contains all the jitter files used in the experiment.
* (Only required if one whishes to use the WoF) A folder named **Wheel\_of\_Fortune** which contains all the scripts required for the WoF.

**All images** need to be placed in the same folder as EAT\_main.m to be loaded. They **do not** go into a separate images folder. The images can be downloaded with the most current directory.

## *Instructions*

All the instructions that have numbers in them that update according to values that are updated within the paradigm (e.g. seconds) are still hardcoded within EAT\_main.m. This is an issue that is planned to be resolved in the next update. All instructions that do not take input during the paradigm are taken from instruction scripts, such as tVNSGerman.m or BEDVARGerman.m.   
In these instruction scripts, the instructions are saved in the structure *instr.* Throughout the script, the DrawFormattedText function calls on the different strings in this structure to print out the instructions on the screen. Therefore, any new script should follow the names of the strings used in the existing language files.

## *Jitters*

New jitter files can be made with Computejitter\_exp.m. This function takes 3 arguments:

1. Mu\_jitter: the average value for the jitter
2. Max\_delay: the longest length for the jitter
3. N\_trials: the number of trials for which the jitter is made

If you want to use a new jitter combination, make sure to run the function with the values corresponding to your experiment, and save the resulting .mat file in the *jitters* folder.

In order to load your jitters, make sure to fill out the appropriate jitter values in the jitter ‘timings’ section when creating the structure with experiment settings, or when filling it out in the main script (see Part 2: Timings).

## *Training*

The training file is a .m file (TrainEAT\_main.m) that is called by the main file, to run the evaluation of the maximum effort. The first part of the script runs this estimation trials, the second part of the script runs a *bidding phase.* In both phases the structure of the code is very similar to EAT\_main.m, using the same procedures for ball movement unless otherwise indicated in the description below.

### Estimation trials

These are 2 or 3 trials (depending on the settings, see *Part 1: Settings*.) in which the participant is prompted to exert either maximum or minimum effort (just holding the device). In the case of 2 trials, we recommend using both to estimate the maximum. In the case of 3 trials the middle trial can be used for minimum effort estimation. This is only necessary if the device has an arbitrary minimum, like the grip force device that we are using to operate the paradigm. The execution of this section is like the execution of the main script, the main difference being that the bar in the tube moves upward, to indicate the maximum effort reached, rather than the minimum effort required to earn points. If the bidding phase is turned off, this part of the script will directly flow back to the main script to continue with the rest of the training, using the same code as the actual experimental run.

Relevant output is collected in the ‘collectMax’ structure, and includes the minEffort and maxEffort, used to determine the ball specifications in the following ‘official’ trials. It also contains the effort data from each of the estimation trials.

### Bidding phase

This bidding phase can be turned on in the settings of the main script. It functions to make the food-points and the money-points worth about the same, which can be in different ratio’s depending on the individual. This money-food ratio is can’t grow larger than 1:2 or smaller than 1:0.5. The value of the food points is variable, depending on the effort during the bidding phase and the (invariable) value of money points, for which the value is set at the beginning of the main script.

The Bidding phase runs a total of 6 trials, in which food and money are presented at different sizes: 50, 100 or 200 cents or kcal, depending on the condition. At the end of the three trials, the function polyfit is used to estimate how similar the desirability of food and money is. The resulting coefficient (saved as input\_device.value\_factor) indicates the disruption between the desirability: values close to 1 indicate wanting is similar for money and food, lower values (up to 0.5) indicate that food is more desirable and higher values (up to 2) indicate that money is more desirable.

Relevant output is collected in the ‘collectBid’ structure, and includes the effort data of each of the bidding files.

## *VAS*

VAS is run with the extra script: Effort\_VAS.m. This script takes the string in trial.question (e.g. ‘wanted’) and places the appropriate question above a scale with a moveable red bar. For ‘wanted’ and ‘exhausted’ the questions are predefined :

Wanted: *‚Wie sehr wollten Sie die Belohnung in diesem Durchgang erhalten?‘*

Exhausted: *‚Wie stark haben Sie sich in diesem Durchgang verausgabt?‘*

For all other input in trial.question, the text displayed is as follows:

*'Wie ' trial.question ' fuehlen Sie sich im Moment?'*

Regardless of the trial question, the scale starts (on the left) with ‘Überhaupt nicht’ and ends (to the right) with ‘Sehr’.

The relevant output of Effort\_VAS.m are:

* startTime (start time of rating)
* t\_rating\_ref (time of rating submission)
* rating (rating value)
* rating\_label (trial.question)
* rating\_subm (1 if rating submitted through user confirmation (e.g. mouse click), 0 if rating is taken at end of time limit without user confirmation)

## *WoF/PANAS*

Pending.

# **References**

Meyniel, F., Safra, L., & Pessiglione, M. (2014). How the brain decides when to work and when to rest: dissociation of implicit-reactive from explicit-predictive computational processes. PLoS Comput Biol, 10(4), e1003584. doi:10.1371/journal.pcbi.1003584 Meyniel, F., Sergent, C., Rigoux, L., Daunizeau, J., & Pessiglione, M. (2013). Neurocomputational account of how the human brain decides when to have a break. Proc Natl Acad Sci U S A, 110(7), 2641-2646. doi:10.1073/pnas.1211925110

Neuser, M. P., Teckentrup, V., Kühnel, A., Hallschmid, M., Walter, M., & Kroemer, N. B. (2019). Vagus nerve stimulation increases vigor to work for rewards. bioRxiv, 789982.

1. EAT versions are named after the Miffy books, that have been part of lab discussion around the time of the development of the larger task [↑](#footnote-ref-1)