AttMem

A Whisker client

by Rudolf Cardinal

www.whiskercontrol.com

Distributed by Campden Instruments Ltd (www.campden-inst.com)

AttMem

Printed: February 2018 in Cambridge, UK

Creator (Whisker)

Rudolf N. Cardinal

Design and Programming (Whisker)

Rudolf N. Cardinal

Michael R. F. Aitken

Legal Advisor (CUTS)

Adjoa D. Tamakloe

Sales (Campden)

Julie Gill

Contacting the authors:

For information about Whisker, visit http://www.whiskercontrol.com/.

If you have sales enquiries about Whisker, contact Campden Instruments Ltd at http://www.campden-inst.com/.

If you have comments or technical enquiries that cannot be answered by the sales team, contact the authors:

Rudolf Cardinal (rudolf@pobox.com)
Mike Aitken (m.aitken@psychol.cam.ac.uk)

Table of Contents

Foreword

WARNING

Whisker is a system designed for research purposes only, and should never be used to control medical apparatus or other devices that could endanger human life.

DISCLAIMER

The authors, copyright holders, and distributors disclaim all responsibility for any adverse effects that may occur as a result of a user disregarding the above warning.

1 AttMem

1.1 About AttMem

Purpose

Attention and working memory task.

Software requirements

Requires Whisker v2.0 or greater.

Data storage

- Text-based output to disk.
- ODBC data storage to a database (supplied).

Author

Rudolf Cardinal (rudolf@pobox.com).

Copyright

Copyright © Cambridge University Technical Services Ltd

Version history

- See version tracker (online).
- Version 4.0 (Dec 2009): XML configuration files; ability to punish premature responding in Phase 1; multiple options for stimulus duration and pre-stimulus pause, and draw-without-replacement multiplier for delays; database table rename (for lookup table) and field changes to deal with these parameters; more thorough recording of all (e.g. panel-push) responses, even not very interesting ones; textfile filename/session changes automatically on config reload; better help.
- Version 4.1 (24 Apr 2010): (a) bugfix: timer left dangling (problem if subject responded *during* phase 1 stimulus, I think; symptom was Phase 2 stimulus being curtailed); (b) ensure limited holds exceed stimulus durations; (c) distinguish (in phases) stimulus 1/2 "on" versus "done".
- Version 4.3 (14 Sep 2010): (a) database query addition to look at target/distractor holes more clearly; (b) option to enable/disable individual holes.

1.2 Required devices

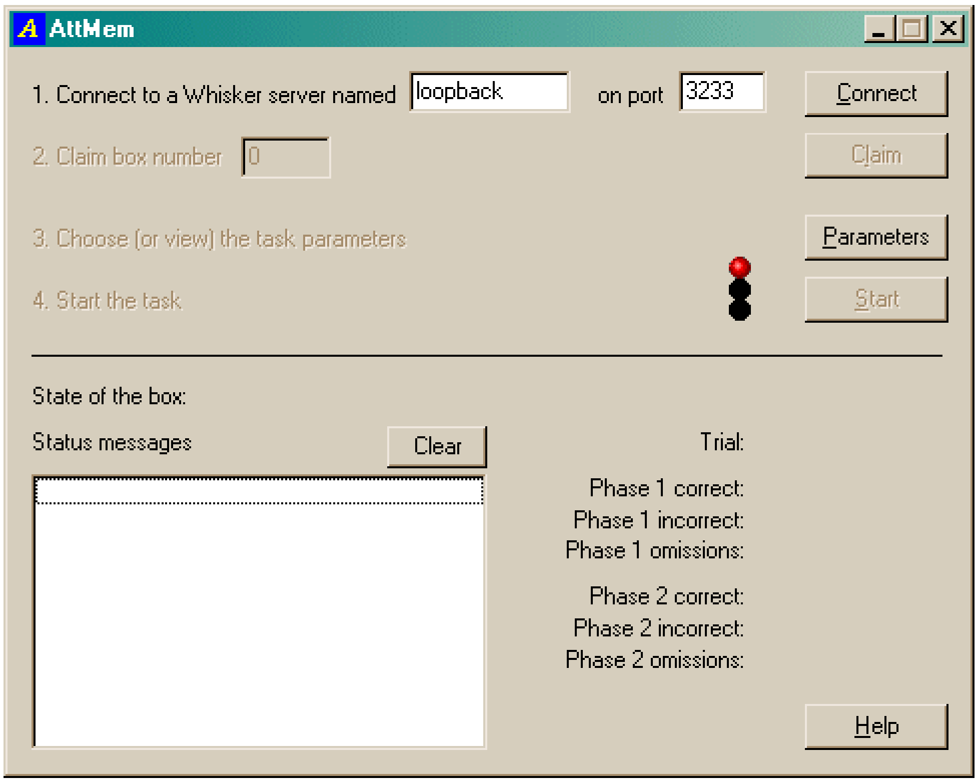
The program requires to claim devices in groups named **box0, box1, box2...** with device names as listed below in bold:

```
# ---------------- Box 0 definition
# INPUTS
line    0       box0    REARPANEL
line    3       box0    HOLE_0
line    6       box0    HOLE_1
line    9       box0    HOLE_2
line    12      box0    HOLE_3
line    15      box0    HOLE_4

# OUTPUTS
line    24      box0    HOUSELIGHT
line    27      box0    PELLET
line    30      box0    TRAYLIGHT
line    36      box0    STIMLIGHT_0
line    39      box0    STIMLIGHT_1
line    42      box0    STIMLIGHT_2
line    45      box0    STIMLIGHT_3
line    48      box0    STIMLIGHT_4

# ... and so on
```

Please ensure that these devices are available and listed in the device definition file in use by the server. (The snippet above shows an extract from a typical definition file.)

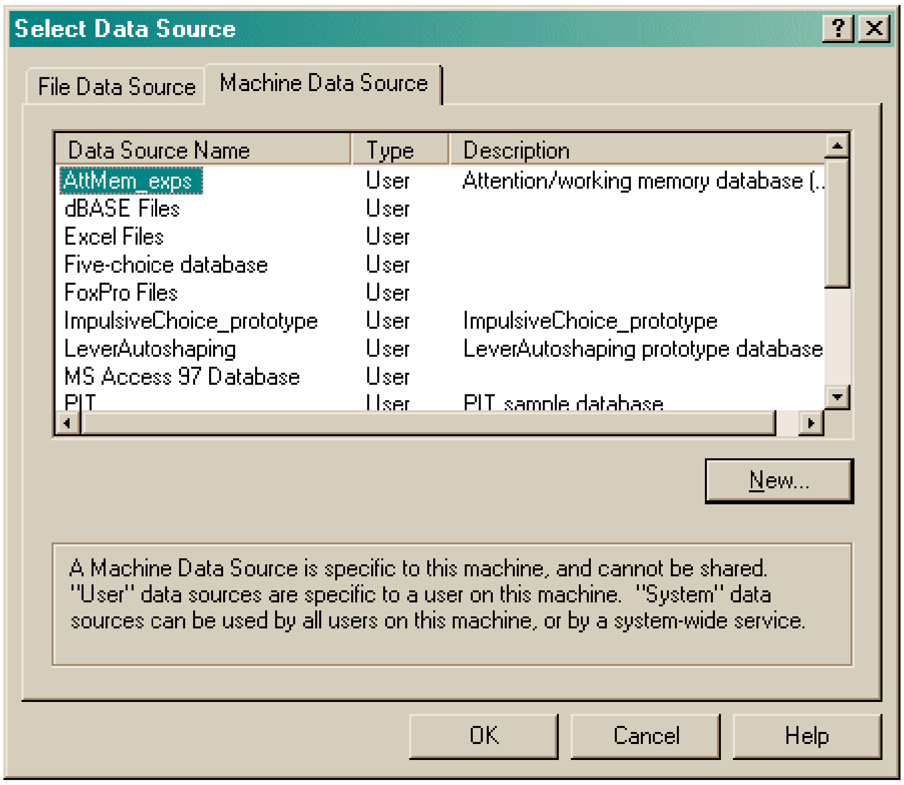
1.3 Using the task

When you run the task, the main screen looks as follows:



You must connect to a Whisker server, claim an operant chamber (box), and set up the parameters for your task. Once that's done, the traffic lights will turn amber. When you are ready, press *Start* to begin the task.

When the task finishes, it saves data to disk and pops up a new dialogue box for you to select a database to store the data to. (The data sources are configured under *Control Panel* → *ODBC*.) If you previously specified an ODBC data source in the parameters, that data source is used automatically and you will only see a dialogue box if something goes wrong and the program needs your input.



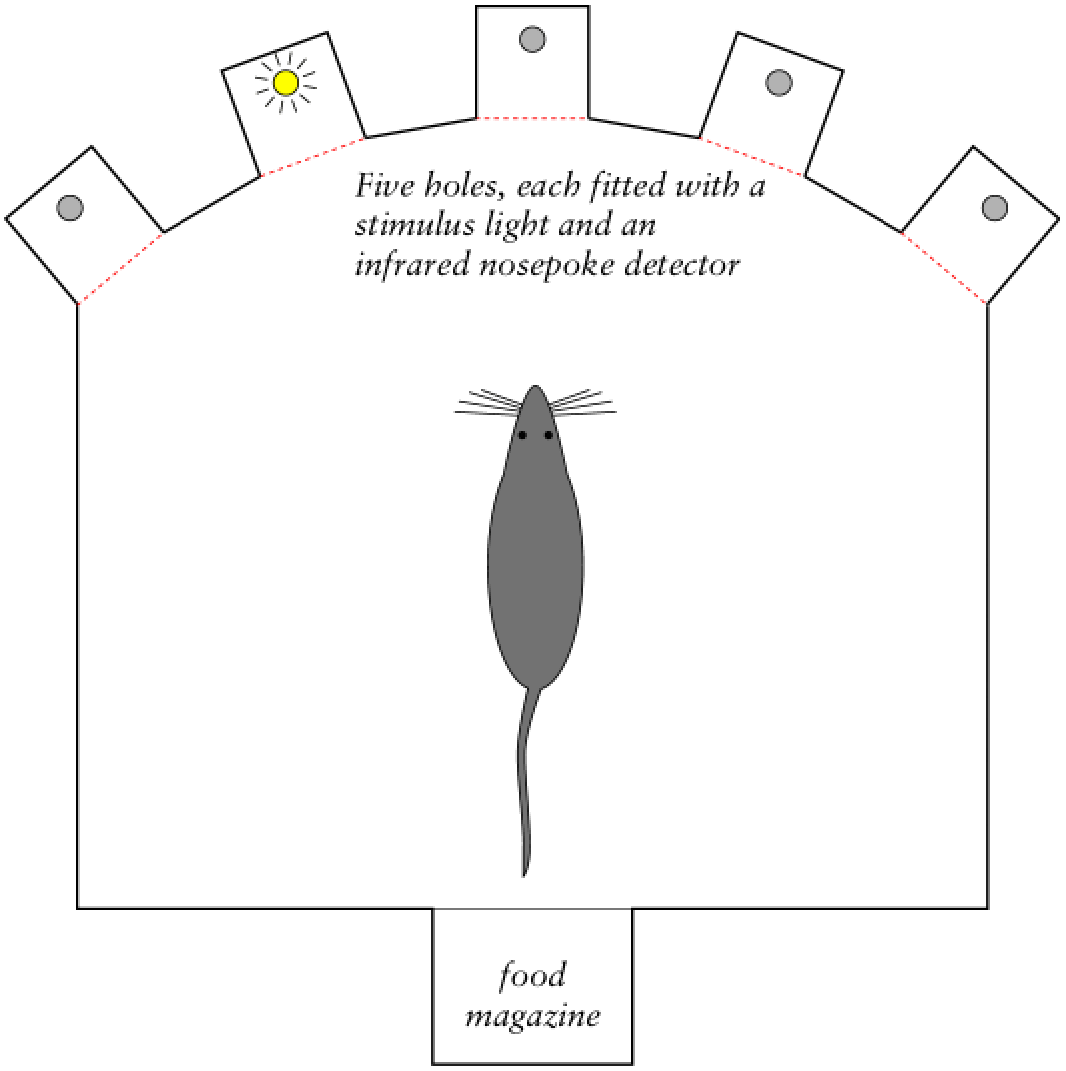
Data Source Name	Type	Description
AttMem_exps	User	Attention/working memory database (..
dBASE Files	User	
Excel Files	User	
Five-choice database	User	
FoxPro Files	User	
ImpulsiveChoice_prototype	User	ImpulsiveChoice_prototype
LeverAutoshaping	User	LeverAutoshaping prototype database
MS Access 97 Database	User	
PIT	User	PIT sample database

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

1.4 Task details

Equipment

The task runs in a standard five-hole box (or nine-hole box using only five holes):


Five holes, each fitted with a stimulus light and an infrared nosepoke detector
food magazine

Task details

The task is a series of *n* trials. Each trial has the following format:

- **Attention phase (phase 1).** Light presented in one of five spatial locations. Wait for rat to detect/acknowledge by nosepoking in that hole. *Success* → delay phase. *Failure within limited hold period (failure to respond, or responding in the wrong hole)* → darkness. (Optionally, *premature responding* also leads to the end of the trial and darkness.)
- **Delay phase.** Darkness.
- **Choice phase (phase 2).** Original stimulus ± one or more distractors presented. In the MATCHING task, rat must choose the location of the previous stimulus (*success* → reward; *failure* → end of trial and darkness). In NON-MATCHING task, these contingencies are reversed (i.e. responding to any hole that was illuminated, but was not the Phase 1 hole, is rewarded).
- **Intertrial interval (ITI).** Darkness.

There is an overall *session time limit.* If this expires, the ongoing trial isn't aborted but the task finishes when the current trial is done.

Trial format in detail

Think of the task as existing in a number of *states*, because that's how the program implements it. Incidentally, this is an exercise in mental discipline: if you can't specify a state table like this, having thought about what should happen in response to all possible events in any state, then you can't program the task properly.

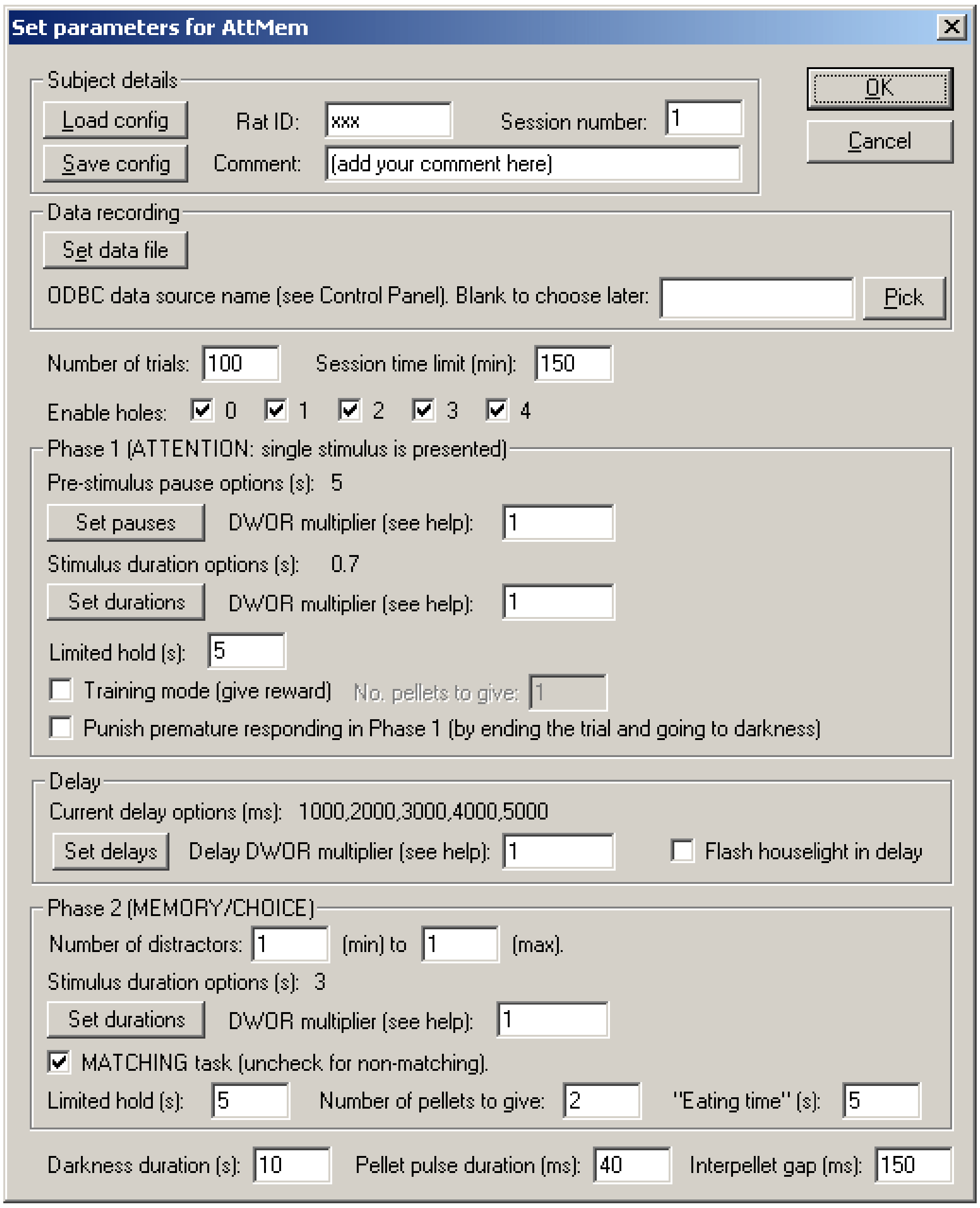
Blue text shows the states that a successful subject passes through.

State name	**Description**	**State of the box**	**Consequence of nosepoking at the front panel (one of the five holes)**	**Consequence of nosepoking at the rear (food) panel**	**Consequence of time passing**
PHASE_NOTSTARTED	"Not started yet". Awaiting experimenter to start task.	Darkness.	-	-	-
PHASE_WANTPANELPUSH	"Awaiting panel-push".	Houselight on.	Recorded. No other consequence.	Recorded. Go to PHASE_INITIAL_PAUSE.	-. *Note potential for long pause.*
PHASE_INITIAL_PAUSE	"Initial pre-stim pause".	Houselight on.	Recorded (anticipatory nosepoke). No other consequence **unless user has chosen to punish these (m_bPunishPhase1Premature),** in which case go to PHASE_DARKNESS.	Recorded (panel push in initial pause). No other consequence.	Go to PHASE_STIM1_ON.
PHASE_STIM1_ON	"Stimulus 1 on".	Houselight on. Stimulus light on.	If correct response, go to PHASE_DELAY (and if rewarding phase 1, also deliver pellets). If incorrect response, go to PHASE_DARKNESS.	Recorded. No other consequence.	Go to PHASE_STIM1_DONE.
PHASE_STIM1_DONE	"Stimulus 1 done, a/w response".	Houselight on. Stimulus off.	Exactly as for PHASE_STIM1_ON.	Recorded. No other consequence.	Limited hold expired; score Phase 1 omission; go to PHASE_DARKNESS.
PHASE_DARKNESS	"DARKNESS... New trial".	Darkness.	Recorded. No other consequence.	Recorded. No other consequence.	Go to PHASE_WANTPANELPUSH.
PHASE_DELAY	"Memory delay".	Houselight on, or if selected, flashing houselight	Recorded (perseverative nosepoke). No other consequence.	Recorded (panel push during delay). No other consequence.	Go to PHASE_DELAYDONE.

		distractor. Traylight on.			
PHASE_DELAYDONE	"Delay finished. Now poke!"	Exactly as for PHASE_DELAY.	Exactly as for PHASE_DELAY.	Recorded (without any special label). Go to PHASE_STIM2_ON.	-. *Note potential for long pause.*
PHASE_STIM2_ON	"Choice stimuli".	Houselight on. Traylight off. Choice stimuli on.	If correct response, give reward and go to PHASE_COLLECTING_BIGBONUS. If incorrect response, go to PHASE_DARKNESS.	Recorded. No other consequence.	Go to PHASE_STIM2_DONE.
PHASE_STIM2_DONE	"Stimulus 2 done, a/w response".	Houselight on. Traylight off. Choice stimuli off.	Exactly as for PHASE_STIM2_ON.	Recorded. No other consequence.	Limited hold expired; score Phase 2 omission; go to PHASE_DARKNESS.
PHASE_COLLECTING_BIGBONUS	"You won. Eat up."	Houselight on. Traylight off.	Recorded. No other consequence.	Recorded. Go to PHASE_EATING_UP.	-. *Note potential for long pause.*
PHASE_EATING_UP	"Munch. Munch. Munch."	Houselight on. Traylight off.	Recorded. No other consequence.	Recorded. No other consequence.	After end of eating time, go to PHASE_DARKNESS.
FINISHED	"--- Finished. ---" Rat finished the task.	Darkness.	-	-	-
ABORTED	"*** Aborted! ***" User aborted the task.	Darkness.	-	-	-

1.5 Parameters

The parameters dialogue box looks like this:



Set parameters for AttMem

Subject details
Load config | Rat ID: xxx | Session number: 1
Save config | Comment: (add your comment here)

OK
Cancel

Data recording
Set data file
ODBC data source name (see Control Panel). Blank to choose later: | Pick

Number of trials: 100 | Session time limit (min): 150

Enable holes: [✓ 0] [✓ 1] [✓ 2] [✓ 3] [✓ 4]

Phase 1 (ATTENTION: single stimulus is presented)
Pre-stimulus pause options (s): 5
Set pauses | DWOR multiplier (see help): 1
Stimulus duration options (s): 0.7
Set durations | DWOR multiplier (see help): 1
Limited hold (s): 5
☐ Training mode (give reward) | No. pellets to give: 1
☐ Punish premature responding in Phase 1 (by ending the trial and going to darkness)

Delay
Current delay options (ms): 1000,2000,3000,4000,5000
Set delays | Delay DWOR multiplier (see help): 1 | ☐ Flash houselight in delay

Phase 2 (MEMORY/CHOICE)
Number of distractors: 1 (min) to 1 (max).
Stimulus duration options (s): 3
Set durations | DWOR multiplier (see help): 1
[✓] MATCHING task (uncheck for non-matching).
Limited hold (s): 5 | Number of pellets to give: 2 | "Eating time" (s): 5

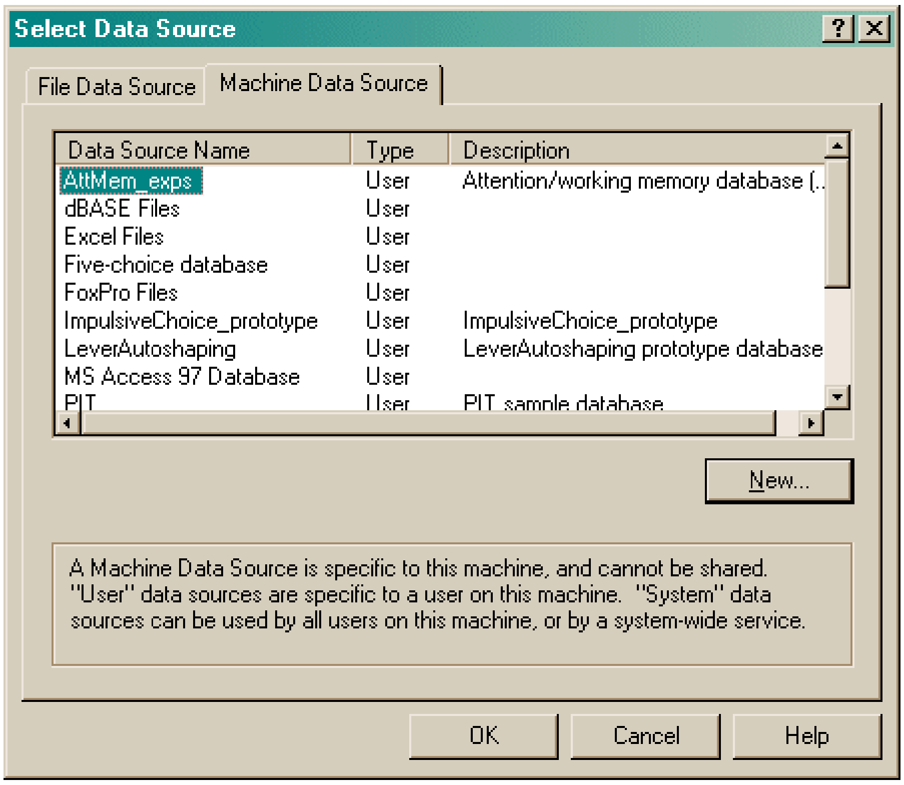
Darkness duration (s): 10 | Pellet pulse duration (ms): 40 | Interpellet gap (ms): 150

Subject details

- **ID, session, comment.** Specify the subject ID, session number (which will auto-advance across successive sessions with the same configuration file) and any comment.

Data recording

- **Set data file.** The task records to (1) a database, via ODBC, and (2) a text file. You may specify the text file manually, or the program will pick a sensible default.
- **Pick database.** To pick an ODBC database **in advance** of finishing, click *Pick* and you will be offered the ODBC Data Source picker (below). Your choice will be recorded and will apply to this subject from now on (or until you specify a different source).



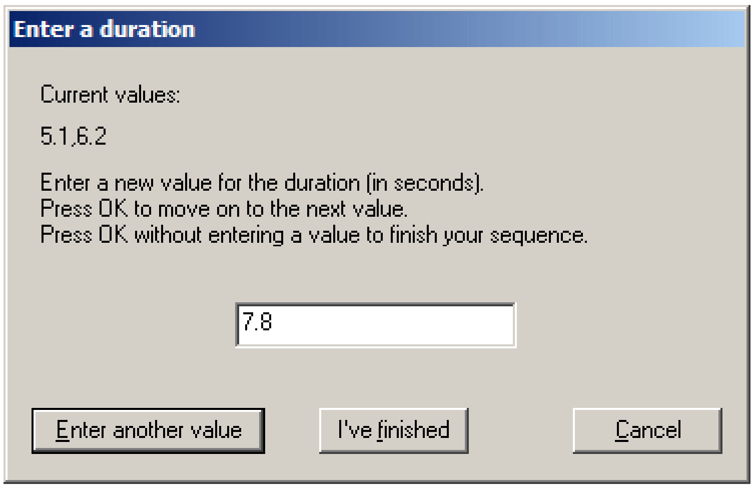
Data Source Name	Type	Description
AttMem_exps	User	Attention/working memory database (..
dBASE Files	User	
Excel Files	User	
Five-choice database	User	
FoxPro Files	User	
ImpulsiveChoice_prototype	User	ImpulsiveChoice_prototype
LeverAutoshaping	User	LeverAutoshaping prototype database
MS Access 97 Database	User	
PIT	User	PIT sample database

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

If you don't specify an ODBC data source now, or you delete the value in the "ODBC data source name" box, you'll be asked to choose when the task ends (and that choice will only apply to the session in progress).

Overall parameters
- **Number of trials.** Specify the overall maximum number of trials.
- **Session time limit.** Specify the overall maximum time limit (in minutes).
- **Enable hole 0/1/2/3/4.** The five holes are numbered 0-4. By default, all are used in the task (all enabled). You can restrict the task to a subset of holes by disabling some. (*Responding to a disabled hole is scored in the same way as responding to a non-illuminated hole, i.e. the holes are disabled for the purposes of presenting stimuli but remain active as inputs.*)

Phase 1 (SAMPLE)
- **Pre-stimulus pause.** The pre-stimulus pause duration (in seconds) is chosen from a list. Specify this list by clicking **Set pauses**. If you only want one value for the delay, create a list with only one entry. The dialogue box for entering durations looks like this:



Enter a duration

Current values:

5.1,6.2

Enter a new value for the duration (in seconds).
Press OK to move on to the next value.
Press OK without entering a value to finish your sequence.

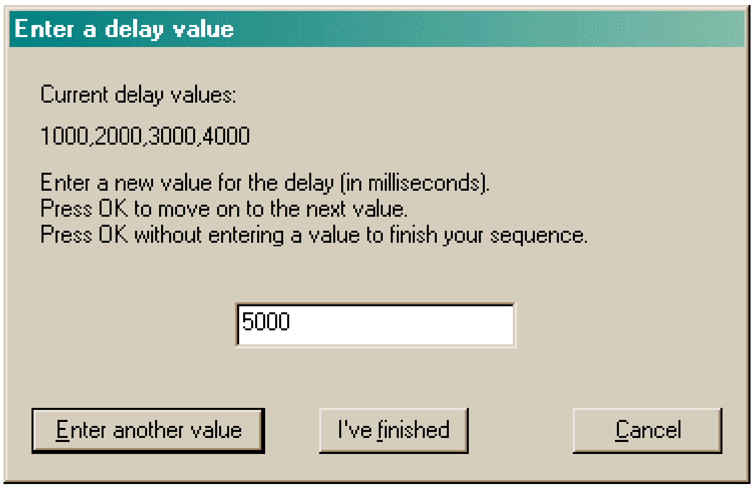
7.8

In addition, the list you entered is "multiplied" up using a **draw-without-replacement (DWOR) multiplier**. This is described in detail here. In brief, suppose your list contains values A and B, like this: {A, B}. If you use a DWOR multiplier of 1, then the task will draw values at random and without replacement from the set {A, B} until it's empty, and then start again. If you use a DWOR multiplier of 3, the task will draw values at random (etc.) from the set {A, A, A, B, B, B} before starting again. Basically, low values (e.g. 1) give pseudorandomness - in a list of length *n*, each value in the list will occur once every *n* trials, but in random order within those *n* trials. As the DWOR multiplier approaches infinity, the behaviour of the task approaches true randomness.

- **Stimulus duration.** Specify the phase 1 stimulus duration list, and DWOR multiplier, likewise.
- **Limited hold.** Specify the limited hold allowed for responding, timed from the start of the phase 1 stimulus.
- **Training mode (give reward).** If this is ticked, then correct phase 1 responding will be rewarded (and you can specify the number of food pellets to be given).
- **Punish premature responding.** If this is ticked, then premature responding (responding at one of the front holes before the phase 1 stimulus has been given) is punished by termination of the trial.

Delay

- **Delay options.** To specify values for the delay, click "Set possible values for the delay":



Enter a delay value

Current delay values:

1000,2000,3000,4000

Enter a new value for the delay (in milliseconds).
Press OK to move on to the next value.
Press OK without entering a value to finish your sequence.

5000

Every time you click "Enter another value", your previous value is added to the list (shown near the top) and you can enter another. A DWOR multiplier is also specified just as described above.

- **Flash houselight in delay.** If ticked, the houselight is flashed on and off during the delay, as a distractor. It's flashed with 250 ms on, then 250 ms off, repeated, for a 2 Hz cycle.

Phase 2 (MEMORY/CHOICE)
- **Number of distractors.** Specify the minimum and maximum number of distractors to be given. On any given trial the actual number is picked at random from within this range.
- **Stimulus duration.** Specify the Phase 2 stimulus duration options, and DWOR multiplier, as above.
- **Matching or non-matching?** Choose whether the matching or non-matching version of the task should be used, as described above (see Task Details).
- **Limited hold.** Specify the phase 2 limited hold period, timed from the start of the phase 2 stimuli.
- **Number of pellets.** Specify the number of pellets to give as a reward if the subject succeeds.
- **Eating time.** Specify the "eating time", from the time the subject collects the reward (nosepokes at the rear food magazine following reward) to entering darkness and an ITI.

Other options
- **Darkness duration.** How long should the ITI (which is spent in darkness) be?
- **Pellet pulse duration.** What length of pulse should be sent to the pellet dispenser to dispense a single pellet? It's usually an electronic, not a mechanical, thing; if you're not sure, try 40 ms.
- **Interpellet gap.** What gap should be left between dispensing separate pellets? This is usually a mechanical, not an electronic thing (too fast and pellets whizz into each other and jam the dispenser); if unsure, try 150 ms or longer.

1.6 Randomness, pseudorandomness, drawing without replacement

Suppose I wish to pick a series of numbers from 1 to 6.

I could pick them **at random**. I could do this by rolling a true die. Every time I rolled the die, I would obtain a number from 1-6 with equal probability. The probability of obtaining a "1" would be 1/6 (approximately 0.17); the probability of obtainin a "2" would be 1/6, and so on.

If I rolled the die 100 times, I would expect to get roughly 17 ones, roughly 17 twos, roughly 17 threes, roughly 17 fours, roughly 17 fives, and roughly 17 sixes. It is *possible* that I would get 100 sixes—but very unlikely (the probability of this is 1.53×10^{-78}, or one in six hundred thousand quintillion quintillion quintillion quintillion). But it is certain that I would not get exactly the same number of ones, twos, threes, fours, fives, and sixes (because you can't have six equal whole numbers adding up to 100). If I rolled the die 60 times, you'd expect about 10 of each number—but it's also pretty unlikely that I'd get *exactly* 10 of each number.

If I rolled the die like this, there is absolutely no way to predict the next number that will come up on each roll.

So much for randomness.

If I want a series of 60 numbers and I want to ensure that I end up with equal numbers of ones, twos, threes, fours, fives, and sixes, I could simply take them in a **predictable order**: 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6. This gives me the distribution I want (ten of each number), but the sequence is anything but random, and is highly predictable. If you know the last number that came up, you have an extremely good idea of what the next number would be.

So much for total predictability.

To obtain a reasonable combination of unpredictability and obtaining an overall equal distribution of numbers, we could use a **pseudorandom** technique called **drawing without replacement**, or

draw-without-replacement, sometimes shortened to **draw-w/o-replacement** or just **DWOR**. Imagine we would like a sequence of 60 numbers, each in the range 1-6, more or less randomly, but ensuring that we get 10 of each number. We could write the number "1" on ten pieces of paper, the number "2" on ten more pieces of paper, and so on. We could then put all 60 pieces of paper in a hat, shuffle them, and draw them out in sequence, *without replacing them*. We'd then be guaranteed ten of each type, but the local order would be fairly random. It would not be totally random—if we've drawn out 10 ones, 10 twos, 9 threes, 10 fours, 10 fives, and 10 sixes, then we can be absolutely certain that the last number out of the hat will be a 3. However, we'd have to remember the numbers that had come out so far.

So it's impossible to have a completely random order and to guarantee a certain distribution—but drawing without replacement is a good way to approximate randomness while guaranteeing a certain distribution.

There are several ways to draw without replacement. I've just described a situation in which 10 copies of each number (1-6) are put into the hat, shuffled, and drawn individually for 60 trials. It's possible (but again extremely unlikely!) that the first ten trials would all be ones, the next ten all twos, and so on. If we wanted to guarantee that *in every six consecutive trials* we will see each possible digit (1-6) once, we should do something different. We should write the number "1" on one piece of paper, the number "2" on a second piece of paper, and so on, up to six pieces of paper. We should then put the six into the hat, shuffle them, and draw them out (without replacement) for the first six trials. We should then put the six pieces of paper back in the hat, shuffle, and repeat the process for the next six trials, and so on. This process gives *less randomness* (if you know just the first five trials in a set of six, then in principle, you can have perfect knowledge of the sixth number to be drawn) but *better control over the local distribution* of numbers.

We've just seen two examples in which a **list** of six numbers (1, 2, 3, 4, 5, 6) are put into a hat and drawn for 60 trials. In the first, we put ten copies of each item in the list into the hat (giving 60 pieces of paper in total) and drew them. I call this a **draw-without-replacement (DWOR) multiplier** of 10. In the second, we put one copy of each item in the list into the hat, drew them until we'd run out of numbers (after six trials), then put them all back into the hat. I call this a DWOR multiplier of 1.

We've just seen the concept of a *list* of possibilities, which is multiplied by a *DWOR multiplier* to give a set of options that are then *drawn at random without replacement* from a hat; when the hat is empty, we restart the process.

The bigger the DWOR multiplier, the closer the DWOR technique comes to total randomness (if the DWOR multiplier were infinitely large, they are exactly the same process). The smaller the DWOR multiplier, the closer the technique is to total predictability.

This program offers the DWOR technique as a way of selecting possible values for various parameters (such as stimulus durations).

Index