INFERNET simulator Reference

For Macintosh computers

© J. Sougné
University of Liège

INFERNET simulator is freeware. It comes with no guaranty.

Caution: INFERNET requires a large amount of memory. The more memory allocated to INFERNET, the better. This application only runs on recent PPC Macintosh computers.

Report bugs and suggestions to J.Sougne@ulg.ac.be

1. Introduction

INFERNET is a connectionist model using integrate-and-fire nodes. In INFERNET, nodes can be in two different states: they can fire ("on"), or they can be at rest ("off"). A node fires at a precise moment and transmits activation to other connected nodes with some time course. When a node activation or potential reaches a threshold, it emits a spike. After firing, the potential is reset to some resting value. INFERNET solves the binding problem by synchrony. Each object is represented by a cluster of nodes firing in synchrony. If the firing distribution is tightly concentrated around the mean, the object is considered to be activated. Objects are bound to their roles by synchronous firing. This synchronous activity defines a window of synchrony: a delay within which the required nodes fire. This delay is constrained by the precision of synchrony (± 5 ms). In INFERNET, discrimination is achieved by successive windows of synchrony. Objects and bindings are maintained in memory by oscillation. Once a node is activated, it tends (but not necessarily) to begin oscillating at a γ (30-80 Hz) frequency range. The temporal gap between 2 spikes of a node is therefore from 14 to 33 ms.

Here follows INFERNET simulator manual. An experiment with INFERNET requires first to define which nodes represent a particular object. The Long term memory should then be defined. The various INFERNET parameters must be set and the simulation can be run. INFERNET provides also a means to analyze INFERNET results.

After double clicking on INFERNET icon the following window appears (Figure 1.1).

Figure 1.1 Introduction window of INFERNET

Click OK and the "Console" window appears (Figure 1.2). This window serves to indicate message and results of various processes.

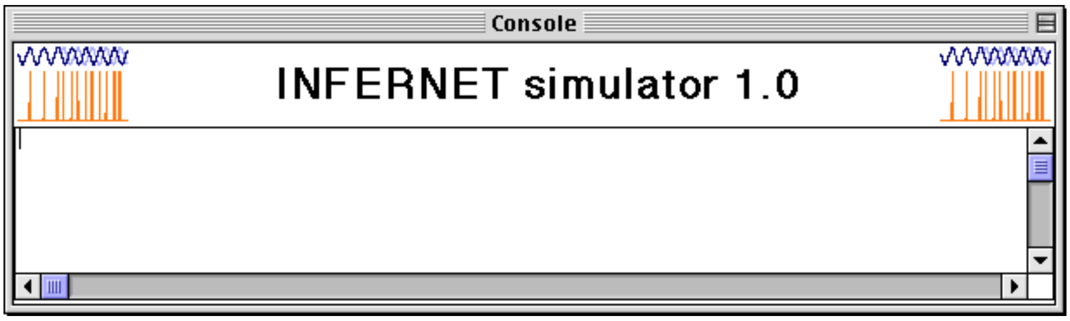
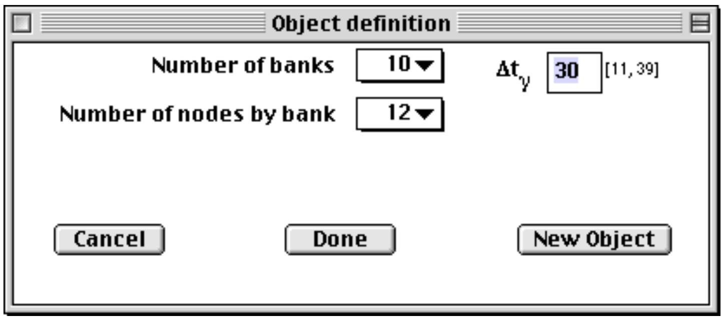


Figure 1.2 INFERNET console window

2. How to set an experiment

2.1 Definition of objects

1. Select the menu item “Define objects” in the Design menu. A window called “Object definition” appears (Figure 2.1). INFERNET is not a fully connected network, its structure is organized by clusters of nodes which constitute sub-nets. Each sub-net is fully connected. From each node of a sub-net there is a connection to every other node in the sub-net. Some of sub-net nodes possess connections to external sub-net nodes. In this window, you must decide how many sub-nets (banks) to use and how many node in each bank. You must also decide the value $\Delta t\gamma$ which is the delay in ms corresponding to a gamma wave. If the oscillation has a frequency of 40Hz, $\Delta t\gamma$ is 25.


Object definition

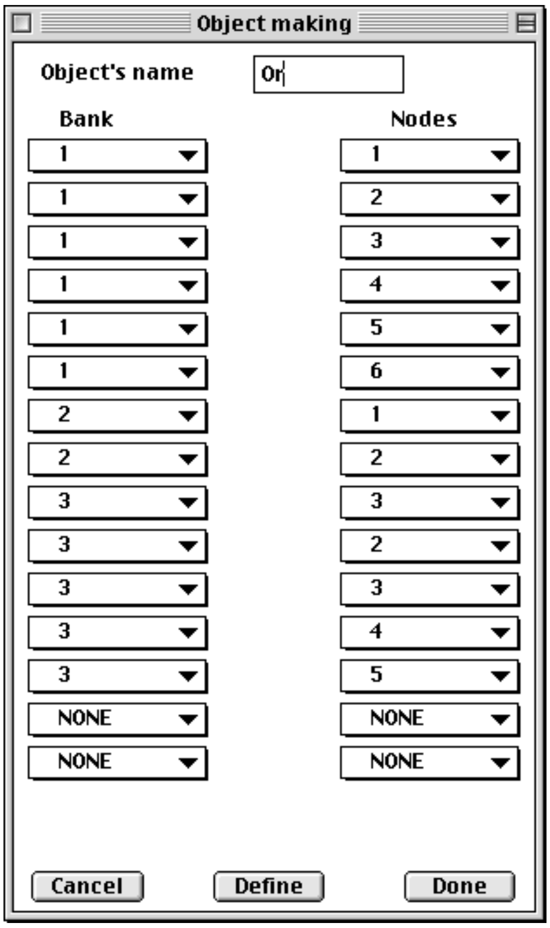
Number of banks: 10

Number of nodes by bank: 12

Δt_γ: 30 [11, 39]

Figure 2.1 Setting of the number of sub-nets nodes and gamma frequency

2. Click on the button “New Object”. The following window appears (Figure 2.2). Here you must give a name to each object and decide which nodes will represent it. Click on the button “Define” to store the object. And define another object. When all objects have been defined click on the button “Done”. The window closes.


Object making

Object's name: Or

Bank	Nodes
1	1
1	2
1	3
1	4
1	5
1	6
2	1
2	2
3	3
3	2
3	3
3	4
3	5
NONE	NONE
NONE	NONE

Figure 2.2 Definition of nodes representing an object

2.2 LTM Definition

1. Select the menu item "Define LTM content" in the Design menu. A window called "LTM building" appears (Figure 2.3). Here you must decide how objects are related. There are six types of connections: post-synaptic excitation, post-synaptic inhibition, pre-synaptic amplification of an excitation, pre-synaptic inhibition of an excitation, pre-synaptic inhibition of an inhibition and pre-synaptic amplification of an inhibition. In the figure, the object "DDD" is connected to the object "ZZZ" with an inhibitory weight (-127), a delay of 10 ms and a full density. It means that all possible connections from "DDD" to "ZZZ" have been set to -127. On these connection there are pre-synaptic inhibitions from nodes representing the object "CCC". It means that when "CCC" nodes fire, they will inhibit, with a 5 ms delay, the inhibition transmitted from "DDD" nodes to "ZZZ" nodes. When the connection has been set, click on the button "Add", and define another connection.

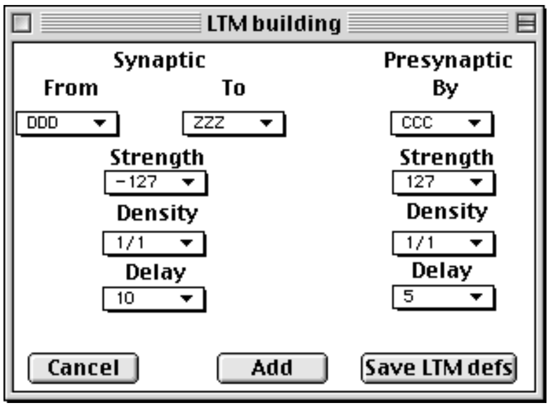
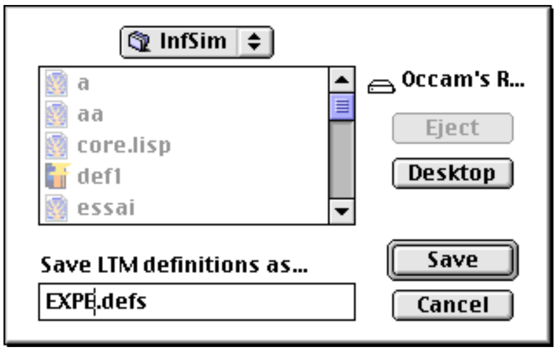


Figure 2.3 LTM definition

2. When all connections have been set, click on the button "Save LTM defs" and the following dialog appears (Figure 2.4). It invites you to save the definitions.

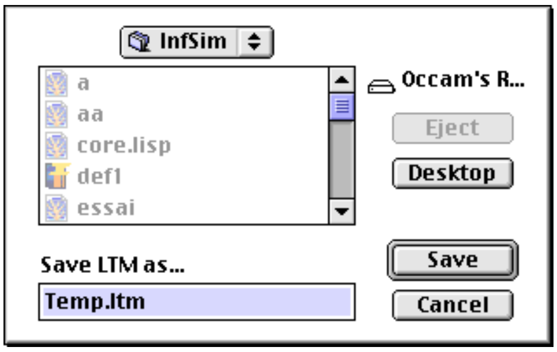


Save LTM definitions as...
EXPE.defs

Figure 2.4 Saving LTM definition

2.3 Save LTM loading file

1. Before using the LTM defined, you must save them in another format. Select the menu item "Make LTM file" in the Design menu. The following dialog appears (Figure 2.5). It invites you to save the LTM.



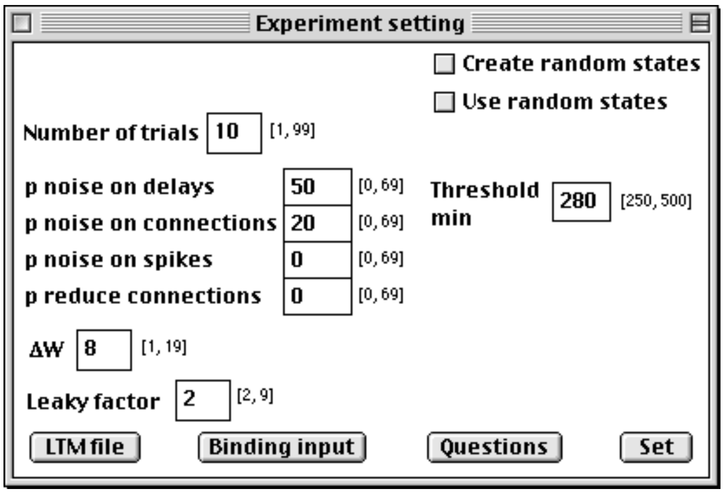
a
aa
core.lisp
def1
essai

Save LTM as... Temp.ltm

Figure 2.5 Saving LTM content

2.4 Set experiment variables

1. Select the menu item “Experiment settings” in the Design menu. A window called “Experiment setting” appears (Figure 2.6). A series of parameters has to be set. Numbers in brackets are minimum and maximum permissible values. All values must be Integers! Choose the number of repetitions of the experiment (Number of trials). Choose the level of noise on transmission delays (p noise on delays). Choose the level of noise on connection weights (p noise on connections). Choose the level of noise on spike (p noise on spikes). Choose the probability to reduce connection weights by noise (p reduce connections). The *Δw* parameter is the learning and forgetting constant. The leaky factor influence the speed of reduction of a node potential. This is the decreasing signal in a Post-synaptic Potential function. The minimum threshold is the minimum level of activation for a node to fire (remember that each connection give a maximum activation of 127). Random states are useful for repeated measure designs. If you want to test different settings while controlling the way random numbers are drawn, using random states is recommended. First create random states, in subsequent experiments check the “Use random states” box.



☐ Create random states
☐ Use random states

Number of trials	10	[1, 99]
p noise on delays	50	[0, 69]
p noise on connections	20	[0, 69]
p noise on spikes	0	[0, 69]
p reduce connections	0	[0, 69]
ΔW	8	[1, 19]
Leaky factor	2	[2, 9]
Threshold min	280	[250, 500]

Figure 2.6 Settings of experiment parameters

2. Click on the button “LTM file” and choose the file to be used in the experiment (Figure 2.7).

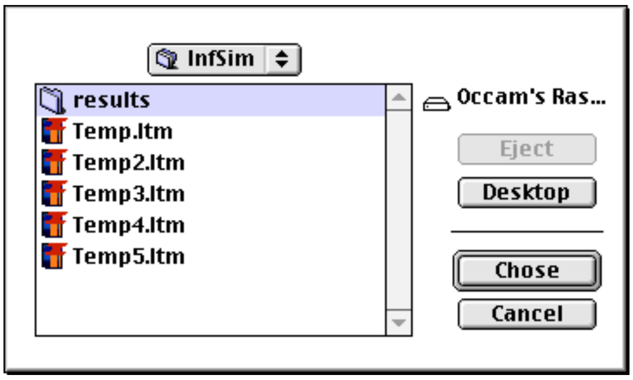
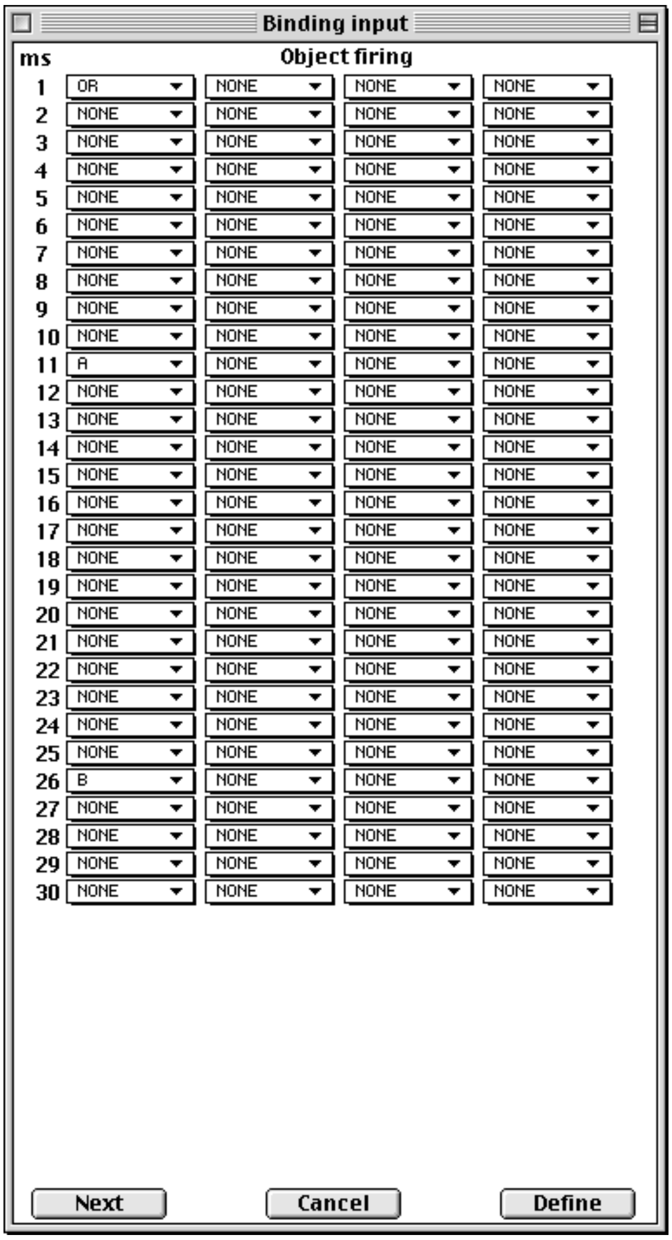


Figure 2.7 Choosing a LTM file for an experiment

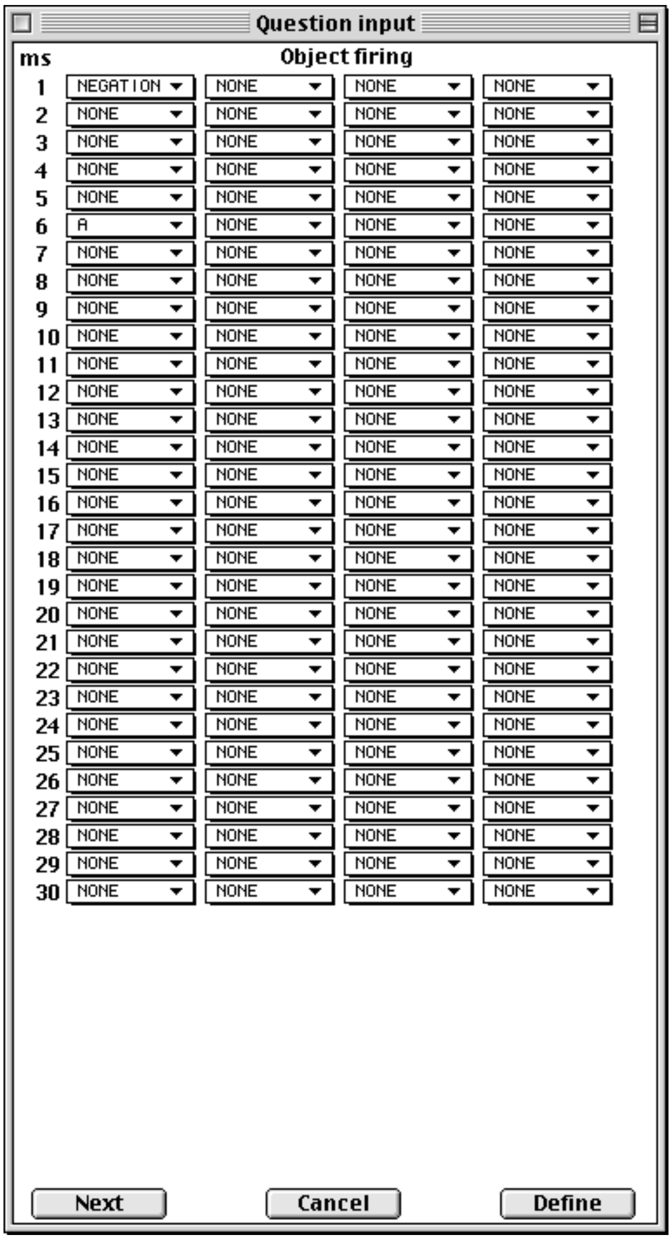
3. Click on the button "Binding input" in Figure 2.6, a new window appears (Figure 2.8). You have to decide which objects nodes will fire at each ms steps. In Figure 2.8, the example shows that this input will be Or (A, B). Be sure to respect delays defined in the LTM files. You have the opportunity to define successive binding inputs by clicking on the button "Next". When all binding inputs are defined, click on the button "Define".



ms	Object firing			
1	OR	NONE	NONE	NONE
2	NONE	NONE	NONE	NONE
3	NONE	NONE	NONE	NONE
4	NONE	NONE	NONE	NONE
5	NONE	NONE	NONE	NONE
6	NONE	NONE	NONE	NONE
7	NONE	NONE	NONE	NONE
8	NONE	NONE	NONE	NONE
9	NONE	NONE	NONE	NONE
10	NONE	NONE	NONE	NONE
11	A	NONE	NONE	NONE
12	NONE	NONE	NONE	NONE
13	NONE	NONE	NONE	NONE
14	NONE	NONE	NONE	NONE
15	NONE	NONE	NONE	NONE
16	NONE	NONE	NONE	NONE
17	NONE	NONE	NONE	NONE
18	NONE	NONE	NONE	NONE
19	NONE	NONE	NONE	NONE
20	NONE	NONE	NONE	NONE
21	NONE	NONE	NONE	NONE
22	NONE	NONE	NONE	NONE
23	NONE	NONE	NONE	NONE
24	NONE	NONE	NONE	NONE
25	NONE	NONE	NONE	NONE
26	B	NONE	NONE	NONE
27	NONE	NONE	NONE	NONE
28	NONE	NONE	NONE	NONE
29	NONE	NONE	NONE	NONE
30	NONE	NONE	NONE	NONE

Figure 2.8 Definition of the binding input

4. Click on the button “Question input”in Figure 2.6, a new window appears (Figure 2.9). You have to decide which objects nodes will fire at each ms steps. In Figure 2.9, the example shows that this input question will be “negation A”. Together with the binding input, this question should enable disjunctive syllogism and produce a “B” response. You have the opportunity to define successive question inputs by clicking on the button “Next”. When all question inputs are defined, click on the button “Define”.



ms	Object firing			
1	NEGATION	NONE	NONE	NONE
2	NONE	NONE	NONE	NONE
3	NONE	NONE	NONE	NONE
4	NONE	NONE	NONE	NONE
5	NONE	NONE	NONE	NONE
6	A	NONE	NONE	NONE
7	NONE	NONE	NONE	NONE
8	NONE	NONE	NONE	NONE
9	NONE	NONE	NONE	NONE
10	NONE	NONE	NONE	NONE
11	NONE	NONE	NONE	NONE
12	NONE	NONE	NONE	NONE
13	NONE	NONE	NONE	NONE
14	NONE	NONE	NONE	NONE
15	NONE	NONE	NONE	NONE
16	NONE	NONE	NONE	NONE
17	NONE	NONE	NONE	NONE
18	NONE	NONE	NONE	NONE
19	NONE	NONE	NONE	NONE
20	NONE	NONE	NONE	NONE
21	NONE	NONE	NONE	NONE
22	NONE	NONE	NONE	NONE
23	NONE	NONE	NONE	NONE
24	NONE	NONE	NONE	NONE
25	NONE	NONE	NONE	NONE
26	NONE	NONE	NONE	NONE
27	NONE	NONE	NONE	NONE
28	NONE	NONE	NONE	NONE
29	NONE	NONE	NONE	NONE
30	NONE	NONE	NONE	NONE

Figure 2.9 Definition of the Question input

5. Click on the button “Set” for terminating experiment setting (Figure 2.6).

2.5 Save an experiment script

The next step is to save a script that contains all experiment definitions. Select the menu item “Save a script” in the File menu. The following dialog appears (Figure 2.10). It invites you to save the script.

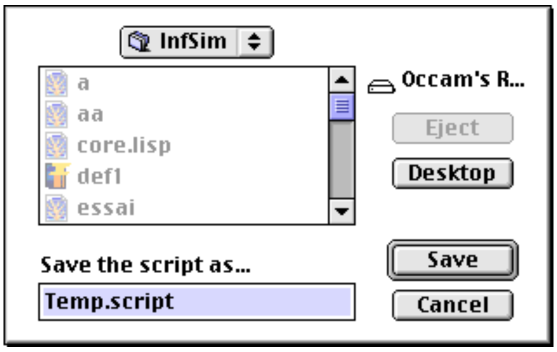


Figure 2.10 Saving a script

2.6 Launch an experiment script

You are ready for running an experiment. Select the menu item “Launch a script” in the File menu. The following dialog appears (Figure 2.11). It invites you to choose the script to be executed. This process may take many hours depending on the size of the network and the number of repetition. It is a good idea to run an experiment only one time to verify that it gives the expected answer before running a full experiment. Caution, when an experiment is running, processor interrupts are blocked, so you won’t be able to use the computer until the experiment finishes!

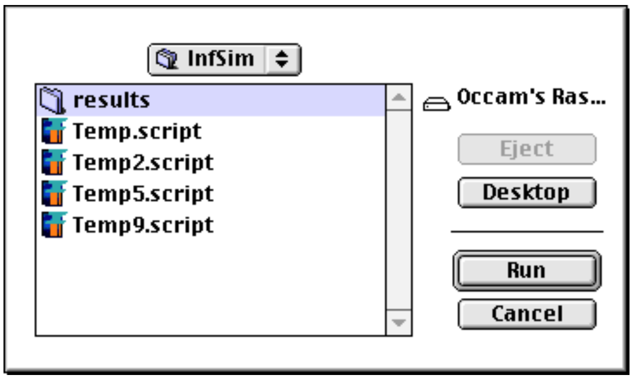


Figure 2.11 Running a script

3. Analysis of results

3.1 Proportion of expected responses and reaction times

3.1.1 Proportion of expected responses

1. Select the menu item “Proportions & RT” in the Analysis menu. A window called “Correction” appears (Figure 3.1).

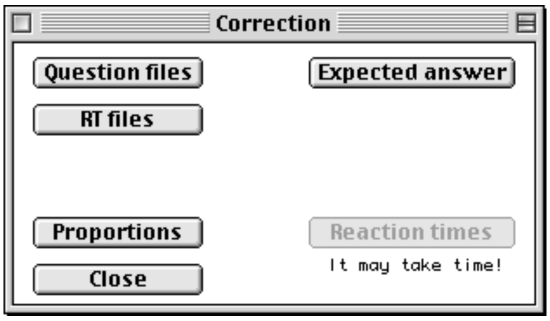


Figure 3.1 Window of correction

2. Click on the Question files button and select one of the question file you want to test (Figure 3.2). If there are many trial, the system will analyze all trials for this question.

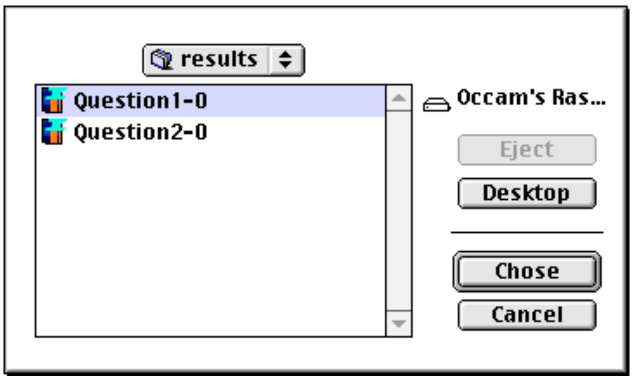
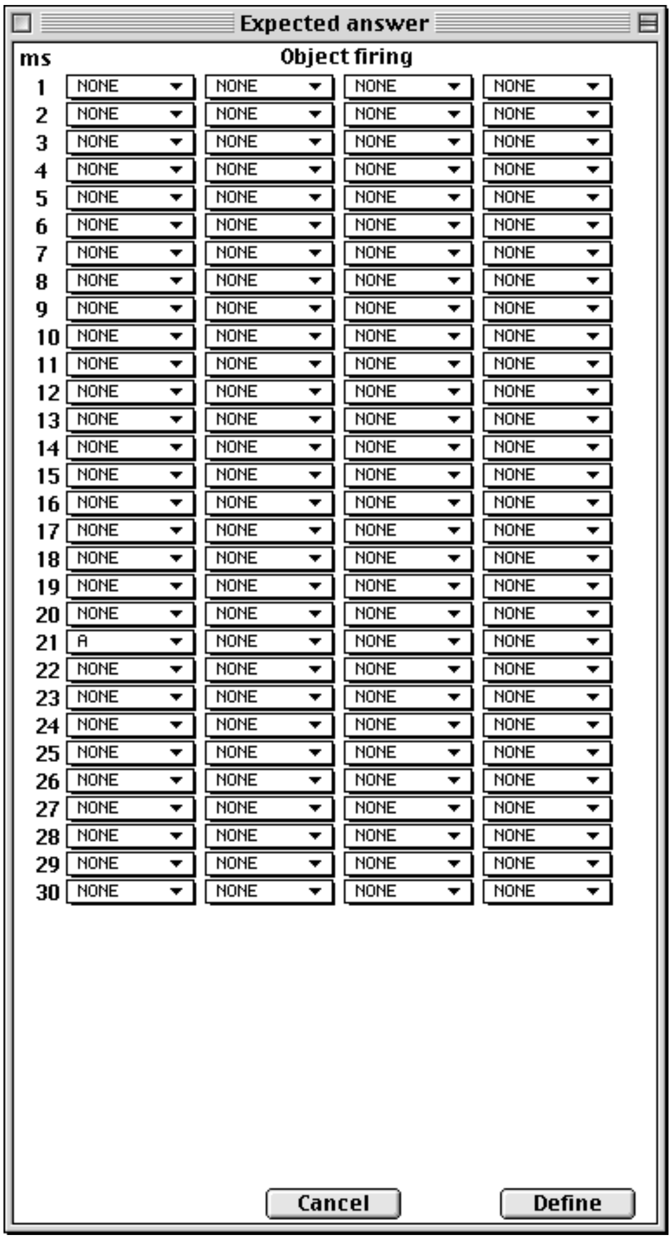


Figure 3.2 Selection of question files

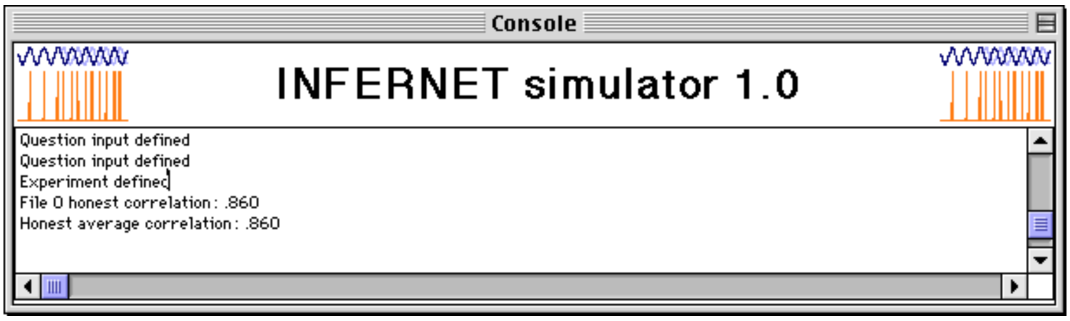
3. Click on the button “Expected answer”, the following window appears (Figure 3.3). Select the objects required firing times. Figure 3.3 shows the expectation of object “A” firing at the 21st ms. Click on the button “Define” and the window closes.



ms	Object firing			
1	NONE	NONE	NONE	NONE
2	NONE	NONE	NONE	NONE
3	NONE	NONE	NONE	NONE
4	NONE	NONE	NONE	NONE
5	NONE	NONE	NONE	NONE
6	NONE	NONE	NONE	NONE
7	NONE	NONE	NONE	NONE
8	NONE	NONE	NONE	NONE
9	NONE	NONE	NONE	NONE
10	NONE	NONE	NONE	NONE
11	NONE	NONE	NONE	NONE
12	NONE	NONE	NONE	NONE
13	NONE	NONE	NONE	NONE
14	NONE	NONE	NONE	NONE
15	NONE	NONE	NONE	NONE
16	NONE	NONE	NONE	NONE
17	NONE	NONE	NONE	NONE
18	NONE	NONE	NONE	NONE
19	NONE	NONE	NONE	NONE
20	NONE	NONE	NONE	NONE
21	A	NONE	NONE	NONE
22	NONE	NONE	NONE	NONE
23	NONE	NONE	NONE	NONE
24	NONE	NONE	NONE	NONE
25	NONE	NONE	NONE	NONE
26	NONE	NONE	NONE	NONE
27	NONE	NONE	NONE	NONE
28	NONE	NONE	NONE	NONE
29	NONE	NONE	NONE	NONE
30	NONE	NONE	NONE	NONE

Figure 3.3 Expected answer setting

4. Click on the button “Proportions” (Figure 3.1), results are displayed in the console window (Figure 3.4).



```
Question input defined
Question input defined
Experiment defined
File 0 honest correlation: .860
Honest average correlation: .860
```

Figure 3.4 Proportions results expressed in correlations

3.1.2 Reaction times

1. Click on the “RT files” button (Figure 3.1) and select one of the Binding file you want to test (Figure 3.5). If there are many trial, the system will analyze all trials.

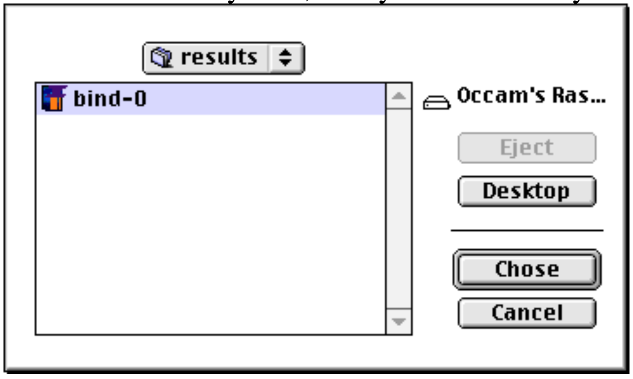
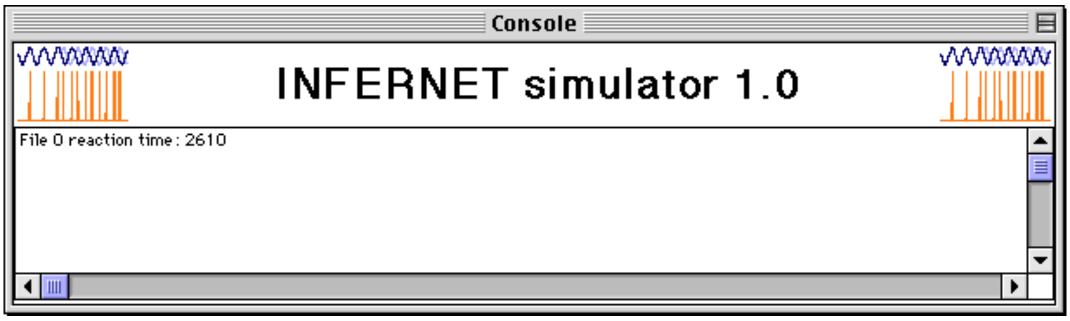


Figure 3.5 Selection of binding files

2. Click on the button “Expected answer”, a window appears (Figure 3.3). Select the objects required firing times.
3. Click on the button “Reaction times” and wait for results displayed on the Console window (Figure 3.6).



```
File 0 reaction time : 2610
```

Figure 3.6 Reaction time result

3.2 Curve of mean theta cycle firing times

1. Select the menu item “Theta trace” in the Analysis menu. INFERNET asks you to select one result file to analyze (Figure 3.7).

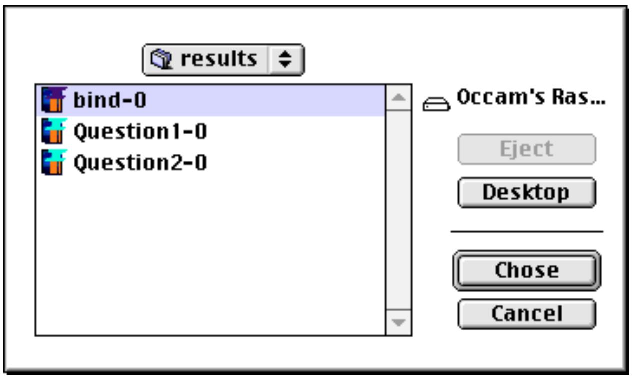
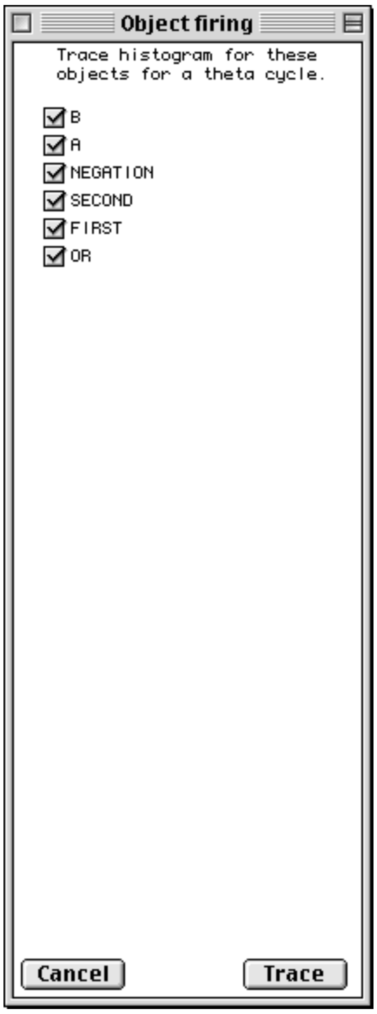


Figure 3.7 Selection of a result file to trace

2. A window called “Object firing” appears (Figure 3.8), select the objects you want to trace and click on the button “Trace”.


Object firing

Trace histogram for these objects for a theta cycle.

☑ B
☑ A
☑ NEGATION
☑ SECOND
☑ FIRST
☑ OR

Figure 3.8 Selection of objects to trace

3. Results appear in a new window called “Object firing density” (Figure 3.9). The curves trace the firing density of nodes pertaining to an object for each ms in theta cycles. On Figure 3.9 we can see that “B” and “SECOND” are synchronized as well as “A” and “FIRST”.

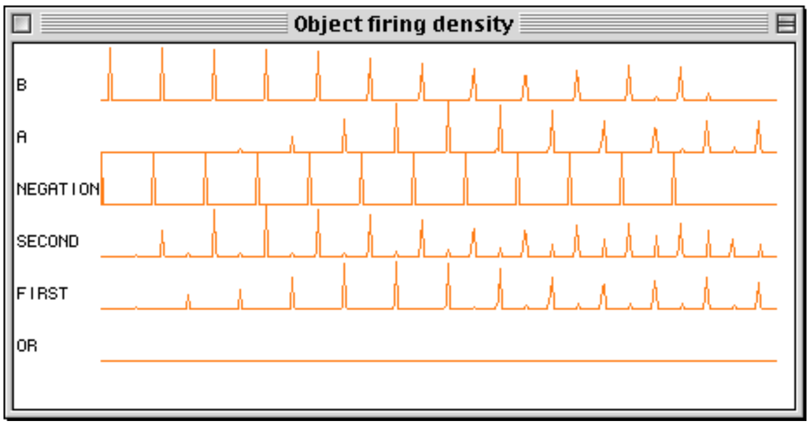


Figure 3.9 Plot of object firing density

3.3 Cross-Correlogram between firing times of two objects nodes

1. Select the menu item “Crosscorrelations” in the Analysis menu. INFERNET ask you to select one result file to analyze (Figure 3.10).

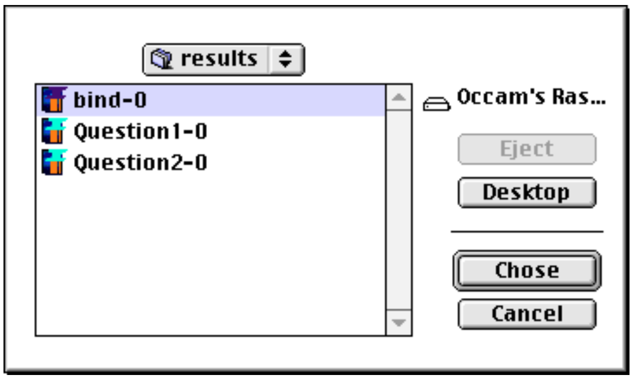


Figure 3.10 Selection of a result file to trace

2. A window called “Correlogram” appears (Figure 3.11). Select the two objects and click on the button “Trace”.

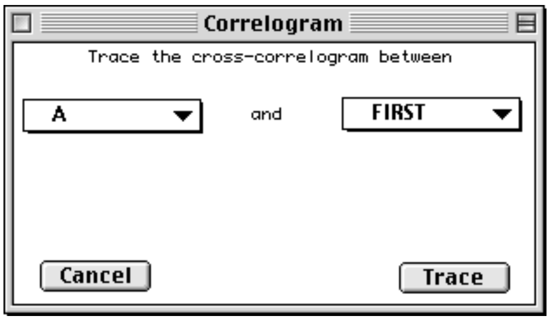


Figure 3.11 Selection of objects to be involved in cross-correlation

3. Results appears in a new window (Figure 3.12).

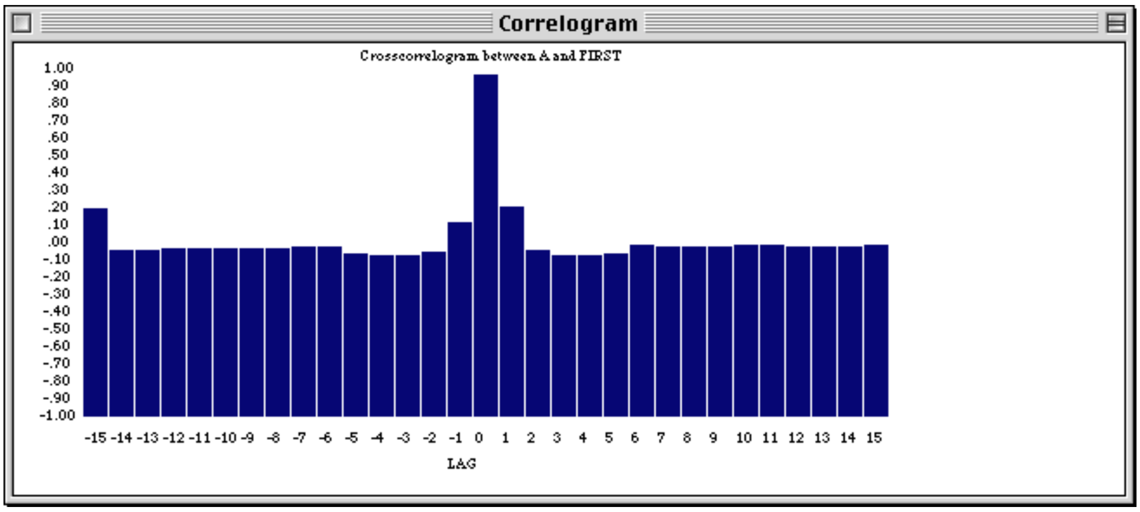
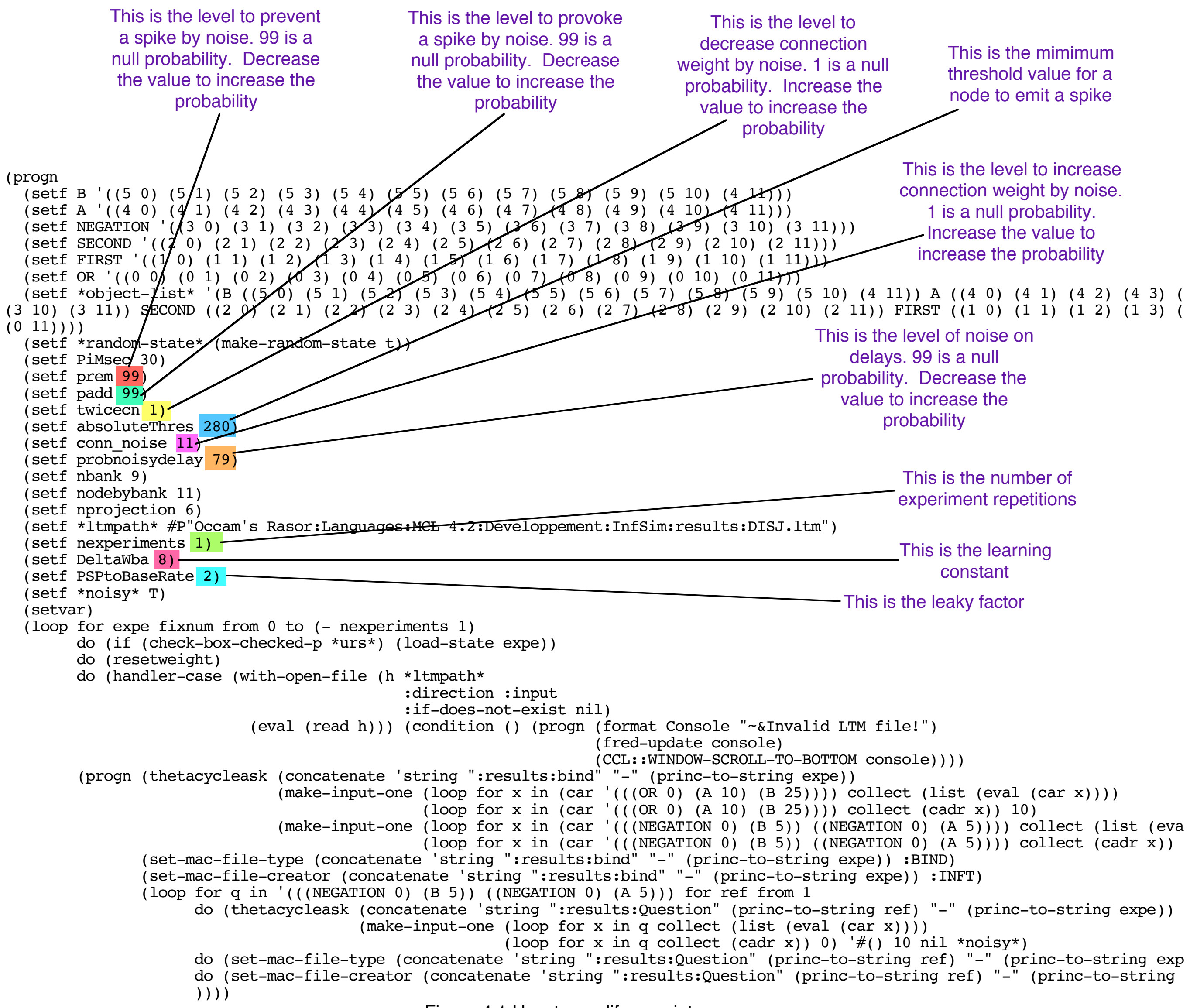


Figure 3.12 Cross-correlogram between 2 objects nodes firing times.

4. Modifying a script by hand

1. Locate the script file and change the file type to TEXT with ResEdit or DataViz FileView™ coming with macLinkPlus.

2. Edit the script with SimpleText or whatever text editor, and modify parameters as indicated in Figure 4.1.


This is the level to prevent a spike by noise. 99 is a null probability. Decrease the value to increase the probability
This is the level to provoke a spike by noise. 99 is a null probability. Decrease the value to increase the probability
This is the level to decrease connection weight by noise. 1 is a null probability. Increase the value to increase the probability
This is the mimimum threshold value for a node to emit a spike
This is the level to increase connection weight by noise. 1 is a null probability. Increase the value to increase the probability
This is the level of noise on delays. 99 is a null probability. Decrease the value to increase the probability
This is the number of experiment repetitions
This is the learning constant
This is the leaky factor

```
(progn
  (setf B '((5 0) (5 1) (5 2) (5 3) (5 4) (5 5) (5 6) (5 7) (5 8) (5 9) (5 10) (4 11)))
  (setf A '((4 0) (4 1) (4 2) (4 3) (4 4) (4 5) (4 6) (4 7) (4 8) (4 9) (4 10) (4 11)))
  (setf NEGATION '((3 0) (3 1) (3 2) (3 3) (3 4) (3 5) (3 6) (3 7) (3 8) (3 9) (3 10) (3 11)))
  (setf SECOND '((2 0) (2 1) (2 2) (2 3) (2 4) (2 5) (2 6) (2 7) (2 8) (2 9) (2 10) (2 11)))
  (setf FIRST '((1 0) (1 1) (1 2) (1 3) (1 4) (1 5) (1 6) (1 7) (1 8) (1 9) (1 10) (1 11)))
  (setf OR '((0 0) (0 1) (0 2) (0 3) (0 4) (0 5) (0 6) (0 7) (0 8) (0 9) (0 10) (0 11)))
  (setf *object-list* '(B ((5 0) (5 1) (5 2) (5 3) (5 4) (5 5) (5 6) (5 7) (5 8) (5 9) (5 10) (4 11)) A ((4 0) (4 1) (4 2) (4 3) (
(3 10) (3 11)) SECOND ((2 0) (2 1) (2 2) (2 3) (2 4) (2 5) (2 6) (2 7) (2 8) (2 9) (2 10) (2 11)) FIRST ((1 0) (1 1) (1 2) (1 3) (
(0 11))))
  (setf *random-state* (make-random-state t))
  (setf PiMsec 30)
  (setf prem 99)
  (setf padd 99)
  (setf twicecn 1)
  (setf absoluteThres 280)
  (setf conn_noise 11)
  (setf probnoisydelay 79)
  (setf nbank 9)
  (setf nodebybank 11)
  (setf nprojection 6)
  (setf *ltmpath* #P"Occam's Rasor:Languages:MCL 4.2:Developpement:InfSim:results:DISJ.ltm")
  (setf nexperiments 1)
  (setf DeltaWba 8)
  (setf PSPtoBaseRate 2)
  (setf *noisy* T)
  (setvar)
  (loop for expe fixnum from 0 to (- nexperiments 1)
        do (if (check-box-checked-p *urs*) (load-state expe))
        do (resetweight)
        do (handler-case (with-open-file (h *ltmpath*
                                            :direction :input
                                            :if-does-not-exist nil)
                         (eval (read h))) (condition () (progn (format Console "~&Invalid LTM file!")
                                                               (fred-update console)
                                                               (CCL::WINDOW-SCROLL-TO-BOTTOM console))))
        (progn (thetacycleask (concatenate 'string ":results:bind" "-" (princ-to-string expe))
                              (make-input-one (loop for x in (car '(((OR 0) (A 10) (B 25)))) collect (list (eval (car x))))
                                              (loop for x in (car '(((OR 0) (A 10) (B 25)))) collect (cadr x)) 10)
                              (make-input-one (loop for x in (car '(((NEGATION 0) (B 5)) ((NEGATION 0) (A 5)))) collect (list (eva
                                              (loop for x in (car '(((NEGATION 0) (B 5)) ((NEGATION 0) (A 5)))) collect (cadr x))
               (set-mac-file-type (concatenate 'string ":results:bind" "-" (princ-to-string expe)) :BIND)
               (set-mac-file-creator (concatenate 'string ":results:bind" "-" (princ-to-string expe)) :INFT)
               (loop for q in '(((NEGATION 0) (B 5)) ((NEGATION 0) (A 5))) for ref from 1
                     do (thetacycleask (concatenate 'string ":results:Question" (princ-to-string ref) "-" (princ-to-string expe))
                                       (make-input-one (loop for x in q collect (list (eval (car x))))
                                                       (loop for x in q collect (cadr x)) 0) '#() 10 nil *noisy*)
                     do (set-mac-file-type (concatenate 'string ":results:Question" (princ-to-string ref) "-" (princ-to-string exp
                     do (set-mac-file-creator (concatenate 'string ":results:Question" (princ-to-string ref) "-" (princ-to-string
                     ))))
```

Figure 4.1 How to modify a script

3. Save the changes.

4. Locate the script file and change the file type with ResEdit or DataViz FileView™ . Be sure the file type is SRIT and the creator INF1.