Online simulations of models for backward masking

Gregory Francis[1]

Purdue University

Department of Psychological Sciences

703 Third Street

West Lafayette, IN 47907-2004

11 July 2002

Revised: 30 January 2003

Final revision: 03 April 2003

Key words: masking, metacontrast, models, simulation, vision

Running head: Online simulations for masking

[1]E-mail: `gfrancis@psych.purdue.edu`; phone: 765-494-6934. This material is based upon work supported by the National Science Foundation under Grant No. 0108905.

Abstract

Five simulations of quantitative models of visual backward masking are available on the Internet at `http://www.psych.purdue.edu/~gfrancis/Publications/BackwardMasking/`. The simulations can be run in a web browser that supports the Java programming language. This paper describes the motivation for making the simulations available and gives a brief introduction to how the simulations are used. The source code is available on the web page, and the paper describes how the code is organized.

Introduction

Backward masking occurs when a briefly presented visual target stimulus becomes difficult to see because of the appearance of a mask stimulus that follows the target. Backward masking has been investigated in thousands of studies with a variety of experimental manipulations (see Breitmeyer & Öğmen (2000) and Enns & Di Lollo (2000) for recent reviews).

There are three reasons that interest in the properties of masking has been strong for decades. First, vision scientists use masking to explore the interaction of the target and mask signals and identify key properties of the mechanisms involved in visual perception. Second, cognitive psychologists use backward masking as a means of interrupting the processing of target information. It is known that processing of a target does not stop with the physical disappearance of the target stimulus, but that processing can continue for at least a second after the stimulus has turned off (Sperling, 1960). The presentation of a strong mask seems to halt further processing of the target stimulus as soon as the mask appears. Thus, by varying the timing between the offset of the target and the onset of the mask, the duration of processing can be controlled and the details of cognitive mechanisms analyzed. Third, the properties of masking have been used to investigate aspects of various types of mental diseases (e.g., Braff & Saccuzzo, 1981; Green, Nuechterlein & Mintz, 1994; Slaghuis & Curran, 1999). Patients sometimes respond quite differently than normals under masking conditions.

Given the strong interest in masking and the frequency of its use as a tool for investigating perceptual, cognitive, and behavioral systems, it is perhaps surprising to note that there is currently no generally agreed upon theory of the mechanisms that are involved in producing masking effects. There is no shortage of theories, but none are generally believed to properly account for the key data in the field. Researchers who use masking as a tool to explore other issues generally have an implicit theory that the mask interrupts processing or interferes with detection of the target properties. However, these ideas are generally not rigorously investigated (usually because the researcher is actually interested in something other than masking *per se*). Even within the field of masking, theories are often only described verbally, and sometimes without a description of the underlying mechanisms that would need to exist to instantiate a theory.

The field of masking would be enhanced if researchers understood and developed quantitative models of masking. A quantitative model is precisely defined and its properties can be demonstrated conclusively. This specificity allows a model to make predictions and for those predictions to be tested experimentally. These tests can, in turn, be used to further the development of the model and so initiate an upward spiral toward the creation of models that account for large amounts of data. In addition, quantitative models can be analyzed mathematically to further our understanding of the basic properties of the models. This latter analysis can help researchers who use masking as a tool to better understand how masking contributes to the primary effect they are studying.

There are several quantitative models of backward masking. In the 1970s, two neural network models were investigated and shown to account for several properties of backward masking. Weisstein (1968, 1972) investigated a dual-channel neural network, while Bridgeman (1971, 1978) studied properties of a neural network with recurrent lateral inhibition. Anbar and Anbar (1982) elaborated a model of brightness perception to show that it accounted for some properties of backward masking. Francis (1997) showed that a neural model of cortical visual processing accounted for many properties of backward masking. Purushothaman, Öğmen and Bedell (2000) showed that a recurrent neural network could produce oscillations that accounted for new data in backward masking. Finally, Di Lollo, Enns and Rensink (2000) showed that a system with reentrant processing could account for their newly discovered object substitution masking effects. A recent analysis by Francis (2000) showed that, despite their differences, many of these models work by the same underlying principles.

Although none of these models account for all of the data on backward masking, many of them use fairly simple principles to account for much of the experimental data. However, the contribution of the quantitative models appears to have not been recognized by many of the researchers that study or use backward masking. For example, Di Lollo *et al.* (2000) reported a new masking effect that they call object substitution. Di Lollo *et al.* suggested that a key property of this new masking effect ruled out existing theories of masking. The key effect was that strong masking could occur for a spatially impoverished mask (four dots around a target) when the target and mask had common onsets but the mask stayed on after the target disappeared. The new effect is quite interesting, and is a new experimental finding; however, Francis and Hermens (2002) showed through computer simulation that some of the quantitative models of masking can account for the effect without a change in parameters. Thus, even mathematically sophisticated researchers (Di Lollo and colleagues have developed their own quantitative models in other contexts) may not recognize the properties of existing quantitative models of backward masking.

The problem is even more severe in fields that use masking as a tool to study other processes. Experimental studies that use masking to investigate some aspect of cognition rarely make reference to any of the models of backward masking. Likewise, clinical psychologists who study the effects of masking in mental disorders never use the quantitative models to explain their findings.

This lack of reference to models is, perhaps, not surprising because the quantitative models are difficult to understand for individuals without a solid background in mathematics. Moreover, in many cases the researcher would need to build a simulation of a model in order to use it to interpret experimental data. Building a simulation is a nontrivial task that can be a source of frustration even for researchers who have received formal training in model simulation. When someone is using masking as an experimental tool to investigate something else, it is perhaps unreasonable to expect them to assign resources toward development of a computer simulation of quantitative models of backward masking.

A recent study by Francis (2000) developed computer simulations of several quantitative models of backward masking. Most of these simulations were written in the Java programming language, so they can run on a variety of computers and they can be easily distributed in web browsers on the Internet. In an effort to promote understanding of the quantitative models of backward masking, I have created a graphical user interface that allows anyone to use these models. The model simulations are available on the web at `http://www.psych.purdue.edu/~gfrancis/Publications/BackwardMasking/`. Use of the models requires a Java-enabled web browser (this includes most web browsers) and access to the Internet. Full source code for the programs is also available as a download from the web page.

The desired effect of making the model simulations widely available is two-fold. First, it is hoped that researchers who want to learn more about quantitative models will use the simulations to help them better understand the properties of the models. There is sometimes a bootstrapping problem with model comprehension, where it is difficult to understand a model without a simulation, but it is nearly impossible to build a simulation without understanding the model. Having an example of a model simulation available might make it easier to understand the model properties. Second, it is hoped that having the simulations readily available will pique the interest of experimentalists to see if the models can account for properties of their data. Such interest may motivate an experimentalist to further explore the characteristics of the models and lead to more interpretations of experimental results with quantitative modeling.

Models available

The models available on the web site include only those models that do not require excessive amounts of computation. In general, a few minutes was considered an upper limit on how long people would wait before getting results from a simulation run. Models that meet this requirement includes those discussed in Weisstein (1972), Bridgeman (1978), Anbar and Anbar (1982), Francis (2000), and Di Lollo *et al.* (2000). Not included are models by Francis (1997) and Purushothaman *et al.* (2000) because these models often must run for hours or days to generate results of interest.

A Java applet resides on the web page to display buttons with names for the different models. Clicking on one of the buttons will cause a new window to appear with a graphical user interface that allows the user to modify stimulus parameters and model parameters. For example, Figure 1 shows the window that appears after clicking on the *Reentrant processing* button. (All screen shots were created while running the simulations on a computer running Mac OS X, but the programs also work on MS Windows and Unix systems with web browsers that support Java.)

- Figure 1 -

Each simulation has the same type of GUI. The window is given a title that indicates which model is being simulated. The left side of the window contains information about an independent variable. This is the variable that is systematically varied during a simulation run. After every run of a simulation, a graph is shown that plots the model's measure of the target percept strength as a function of the individual variable. Target percept strength is a model measure that corresponds to the visibility of the target. In different experiments it would correspond to target brightness, percent correct detections, percent correct discriminations, and so on. The right side of the window lists each model parameter and stimulus parameter that can be set by the user. The bottom of the window lets the user set details of the simulation run. The checkbox labeled *Clear plot* determines whether the next simulation run should be displayed with previous simulation runs or if it should generate an entirely new data plot. The text field labeled *Simulation name* allows the user to enter a name for the next simulation run. This name appears as a label in the legend of the data plot. Finally, clicking on the *Start simulation* button starts the simulation for the given set of parameters. As the simulation proceeds, this button's label is updated to roughly indicate the simulation's progress.

For every parameter, there is a button labeled *Description.* When this button is pressed, a window appears that provides a textual description of what the parameter corresponds to in the model and/or simulation. This information is provided to make it easier for the user to link the parameters in the simulation with the details of the models given in their original publications.

The worth of the simulations can be described by example. The first example will focus on the reentrant processing model of Di Lollo *et al.* (2000) because it is a new model and there is currently substantial interest in the issues raised by Di Lollo *et al.* A key property of the model of Di Lollo *et al.* (2000) concerns the effect of varying the number of distracter items when the target is presented. In the model this is coded by the parameter n. Figure 2a shows the data plot that appears after giving the simulation run the name *n=4* and clicking on the *Start simulation* button. Additional simulation results are added to the graph by changing the parameter n, changing the simulation name, and unchecking the *Clear plot* checkbox. Figures 2b-d show the model output as simulations for other values of n are added to the graph.

- Figure 2 -

Figure 2d is a replication of the results reported by Di Lollo *et al.* (2000) in their Figure 15A. It shows the effect of varying mask duration for different numbers of distracters in the target field. The main result is that there is an interaction of set size and mask duration. The curves in Figure 2 look slightly different than the curves in Figure 15A of Di Lollo *et al.*, but this is because the values of mask durations are slightly different in the two sets of simulations.

Because the graph generated by the program is not of publication quality, the program also provides the simulated data in a plain text format. A menu option called *Show Data* is available on each plot window. Selecting the *Show data points* menu item displays a new window that lists the data in a textual format, as in Figure 3. These data can be selected, copied, and pasted into other programs (e.g., word processor, spreadsheet, text editor). Because columns are tab-delimited, pasting the selected data into a spreadsheet should cause each number to be placed in its own cell. The data can thus be ported into the user's favorite plotting program to create higher quality graphs. The data an also be ported into a statistical analysis program for comparison to experimental data.

- Figure 3 -

The default parameters when a simulation is started always correspond to the parameters used in the original description of the model. However, a user can change these parameters to explore the abilities of the model in other situations. For example, although the reentrant model of Di Lollo *et al.* (2000) was originally designed to account for effects of mask duration with common onset masking and variability in the number of distracters with the target, a researcher might be interested in seeing how it responds to variations in the SOA between the target and mask. A common experimental finding is that if the mask stimulus is weak relative to the target stimulus, the mask has its biggest effect at an intermediate SOA value. When target percept strength is plotted against SOA, the curve takes a u-shape. Accounting for the u-shaped masking function is considered a primary issue in the field of backward masking (Breitmeyer & Öğmen, 2000; Francis, 2000).

Figure 4a shows the behavior of the model when the independent variable is SOA instead of mask duration. All the other parameters were the default values when the simulation started. The model predicts modest masking for common onset of the target and mask and no masking for other SOAs. This finding does not provide evidence against the model of Di Lollo *et al.* (2000) because it was designed to explain other data sets. Interestingly, with a change of parameters the model does seem to be able to produce u-shaped masking functions. By setting the target intensity to be 2.0 and $\lambda = 0.95$, the model produces the curve in Figure 4b, which shows the strongest masking occurs at intermediate SOA values.

- Figure 4 -

This simulation finding is interesting because it suggests that the Di Lollo *et al.* (2000) model, which was designed to account for common onset masking, might also be able to explain effects when the mask onset comes after the target offset. Of course, further research is needed to explore this possibility, and a simulation model is now available to support that research.

Similar investigations can be made with the other model simulations. For example, Bridgeman's (1978) model uses recurrent lateral inhibition across a set of cells. The model includes longer time delays for inhibitory signals that must travel longer pixel distances. A researcher might wonder if the time delays are necessary to produce the model's main effect (the existence of a u-shaped masking function). This can be explored in the model by setting the weights for distances greater than one pixel equal to zero. With zero signal being sent for the longer distances, such a system will have no time delays. Figure 5 shows the masking functions produced for the default parameters and for the simulation without time delays.

Both produce a u-shaped masking function, which indicates that the time delays are not critical to producing the u-shaped masking function.

- Figure 5 -

As a final example, Francis (2000) identified a novel method, called efficient masking, for producing a u-shaped masking function. The default parameter values are those used in Francis (2003) to relate the model to experimental data. Figure 6a shows the kind of masking function produced by the model.

- Figure 6 -

A researcher might wonder if the efficient masking model could also account for the properties of common onset masking studied by Di Lollo *et al.* (2000). The answer is yes if one is willing to hypothesize that increases in set size lead to distributed attention which makes the mask signal have a larger impact on the target signal (see Francis & Hermens, 2002). Figure 6b shows simulation results that demonstrate this property.

Source code

Although one can use the simulation graphical user interface to vary all of the model parameters and to vary the stimulus, there are some situations where modification of the source code will be required. For example, one of the parameters in the model of Di Lollo *et al.* (2000) is interpreted as reflecting the rate at which items are searched in a visual display. For some of their simulations this rate was changed within a run of the simulation to indicate pre-cue and post-cue rates. The simulations reported here cannot duplicate this effect because the search rate can only be one value during a run of the simulation. However, it should be relatively easy for a programmer to modify the source code and allow for this possibility. More generally, a researcher may want to modify one of the models, which will require changes to the programming of the simulation. Thus, a discussion of how the source code is designed may be worthwhile.

The simulation programs are written in the Java programming language. They can be run as stand alone programs or, as has been discussed so far, in a web browser over the Internet. Java is an object oriented language, with each object defined as a class that contains data and methods that manipulate the data.

Many of the classes define fairly general objects, for example, *CloseableFrame.java* defines a class that provides a window that closes upon request. Other classes do fairly complex operations. For example, *SimPlotImage.java* takes results from a simulation, draws an image of a data plot, and displays the image in a window.

The classes of interest for researchers who want to elaborate or modify the simulations are usually named after the researcher(s) who created the models (the one exception is for the Efficient Masking model, which is not named after Francis (2000) in order to avoid confusion with a different model described in Francis (1997)). Thus, the classes are: *AnbarAnbar.java*, *Bridgeman.java*, *DiLolloEtAl.java*, *EfficientMasking.java*, and *Weisstein.java*. Each of these classes defines the variables, parameters, and equations that make up the simulation of the model.

Every simulation program has a common design. The Java programming language has a convenient method of organizing similar types of programs using abstract classes. An abstract simulation class was defined and called *Simulation.java*. The abstract class defines variables and methods that must be part of every simulation. The classes that define a particular model extend the abstract class and immediately acquire the data variables defined by the abstract class. Each particular model class must also provide a definition for each method defined in the abstract class. This insures that each particular model has the basic properties of the abstract class.

The advantage to this arrangement is apparent in the design of the graphical user interface (*SimulationGUI.java*). The type of model being simulated need not be specified for the graphical user interface. Since every model is a version of the simulation class, certain variables always exist and certain methods can always be called. Thus, if a new model simulation is created and it is defined as an extension of the abstract simulation class, the graphical user interface will automatically work with the new model simulation.

Thus, to make a new model simulation, it is only necessary to define a class along the lines of one of the existing models. This involves identifying the parameters for the model and identifying the model calculations. Of course, knowledge of the Java programming language is necessary to make these types of changes.

Conclusions

A set of programs that simulates five models of backward masking is available on the Internet. The programs can either be run through a web browser, or the source code can be downloaded, compiled, and run on a local computer. The programs provide a common interface for interacting with each simulation.

The model simulations will assist researchers interested in backward masking to explore the properties of these models and to thereby gain a better understanding of the types of interactions that may be involved in masking. The goal is to allow the properties of the quantitative models of backward masking to be understood and used by more researchers in the field of masking, experimentalists who use masking as a tool to study other aspects of cognition, and psychiatrists who relate masking effects to various mental conditions. By removing the need for these researchers to create their own simulations of the models, perhaps the researchers will be motivated to use quantitative models to generate novel interpretations of experimental data.

References

Anbar, S. & Anbar, D. (1982). Visual masking: A unified approach. *Perception*, *11*, 427–439.

Braff, D.L. & Saccuzzo, D.P. (1981). Information processing dysfunction in paranoid schizophrenia: A two-factor deficits. *American Journal of Psychiatry 138*, 1051–1056.

Breitmeyer, B. & Öğmen, H. (2000). Recent models and findings in visual backward masking: A comparison, review, and update. *Perception & Psychophysics*, *62*, 1572–1595.

Bridgeman, B. (1971). Metacontrast and lateral inhibition. *Psychological Review*, *78*, 528–539.

Bridgeman, B. (1978). Distributed sensory coding applied to simulations of iconic storage and metacontrast. *Bulletin of Mathematical Biology*, *40*, 605–623.

Di Lollo, V., Enns, J. T., & Rensink, R. A. (2000). Competition for consciousness among visual events: The psychophysics of reentrant visual processes. *Journal of Experimental Psychology: General*, *129*, 481–507.

Enns, J. T. & Di Lollo, V. (2000). What's new in visual masking? *Trends in Cognitive Sciences*, *4*, 345–352.

Francis, G. (1997). Cortical dynamics of lateral inhibition: Metacontrast masking. *Psychological Review*, *104*, 572–594.

Francis, G. (2000). Quantitative theories of metacontrast masking. *Psychological Review*, *107*, 768–785.

Francis, G. (2003). Developing a new quantitative account of backward masking. *Cognitive Psychology*, *46*, 198–226.

Francis, G. & Hermens, F. (2002). Comment on: Competition for consciousness among visual events: The psychophysics of reentrant visual processes, by Di Lollo, Enns and Rensink (2000). *Journal of Experimental Psychology: General*, **131**, 590–593.

Green, M. F., Nuechterlein, K. H., & Mintz, J. (1994). Backward masking in schizophrenia and mania: I. Specifying a mechanism. *Archives of General Psychiatry*, *51*, 939–944.

Purushothaman, G., Öğmen, H. & Bedell, H. E. (2000). Gamma-range oscillations in backward-masking functions and their putative neural correlates. *Psychological Review*, *107*, 556–577.

Slaghuis, W.L. & Curran, C.E. (1999). Spatial frequency masking in positive- and negative-symptom schizophrenia. *Journal of abnormal Psychology*, *108*, 42–50.

Sperling, G. (1960). The information available in brief visual presentations. *Psychological Monographs*, *74*, (11, Whole No. 498).

Weisstein, N. (1968). A Rashevsky-Landahl neural net: Simulation of metacontrast. *Psychological Review*, *75*, 494–521.

Weisstein, N. (1972). Metacontrast. In D. Jameson & L. Hurvich (Eds.) *Handbook of sensory physiology* (Vol. 7, No. 4, *Visual psychophysics*). Berlin: Springer-Verlag.

Figure Captions

Figure 1. The graphical user interface for interacting with the reentrant processing model. The user interface is the same for every model.

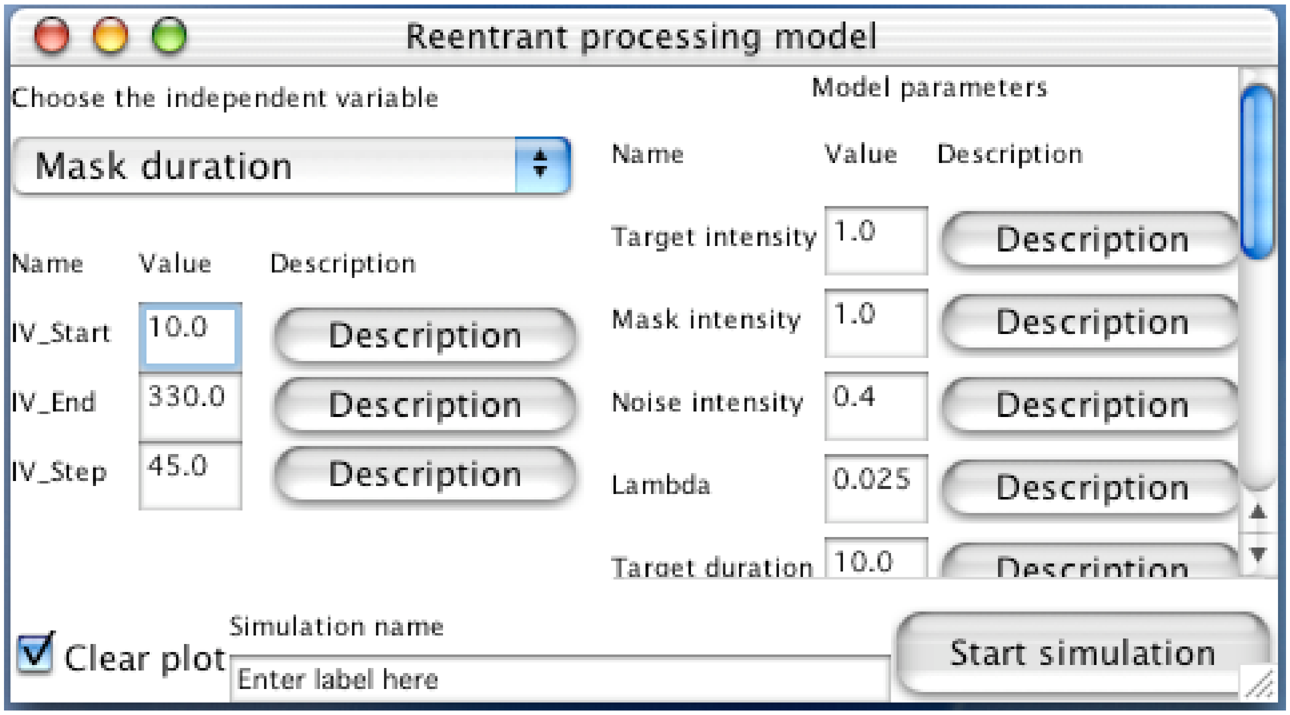
Figure 2. Simulation results for the reentrant processing model. After each run of the simulation, a data plot is produced. Figures (a)-(d) show the build up of plots as new data is added to previous simulation results. The final plot, (d), is a replication of simulation results reported by Di Lollo *et al.* (2000).

Figure 3. To produce better plots or to further analyze the simulated data, selecting the menu option *Show data* from each plot opens a window with a textual listing of the data points. The data can be selected, copied, and pasted into other programs.

Figure 4. Simulation results when the reentrant processing model is tested under new conditions, varying the stimulus onset asynchrony (SOA) between the target and mask stimuli. The results with the original model parameters are shown in (a) and the results with a different set of parameters are shown in (b). The model can produce u-shaped masking functions under some conditions.

Figure 5. Simulation results for the recurrent inhibition model. The two curves differ depending on the weight given to spatial interactions beyond a cell's nearest neighbors. The curve with filled circles marking points is from a simulation with the weights set equal to zero. Setting these weights to zero also has the effect of removing time-delayed inhibition in the model. As the simulations show, u-shaped masking occurs regardless of this property of the model.

Figure 6. Simulation results for the efficient masking model. (a) shows a u-shaped masking function produced by the default set of parameters. (b) shows that the efficient masking model can match the basic properties of object substitution masking (see Figure 2d for comparison).


Reentrant processing model

Choose the independent variable

Mask duration

Name	Value	Description
IV_Start	10.0	Description
IV_End	330.0	Description
IV_Step	45.0	Description

Model parameters

Name	Value	Description
Target intensity	1.0	Description
Mask intensity	1.0	Description
Noise intensity	0.4	Description
Lambda	0.025	Description
Target duration	10.0	Description

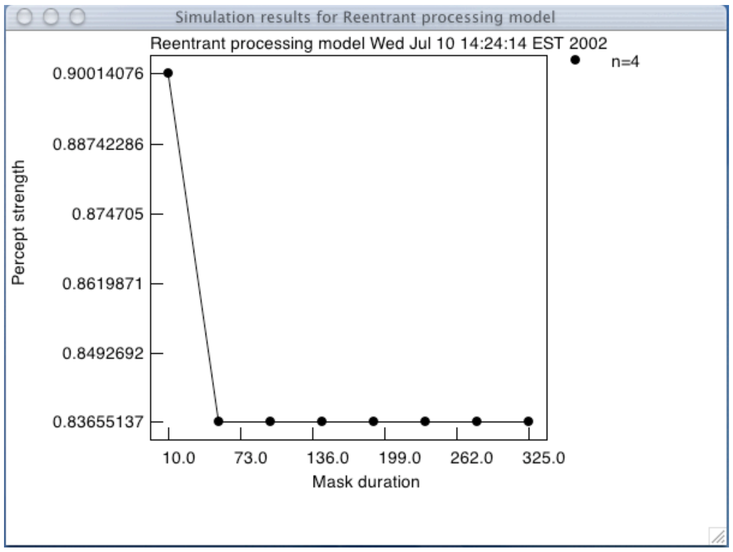
☑ Clear plot

Simulation name: Enter label here

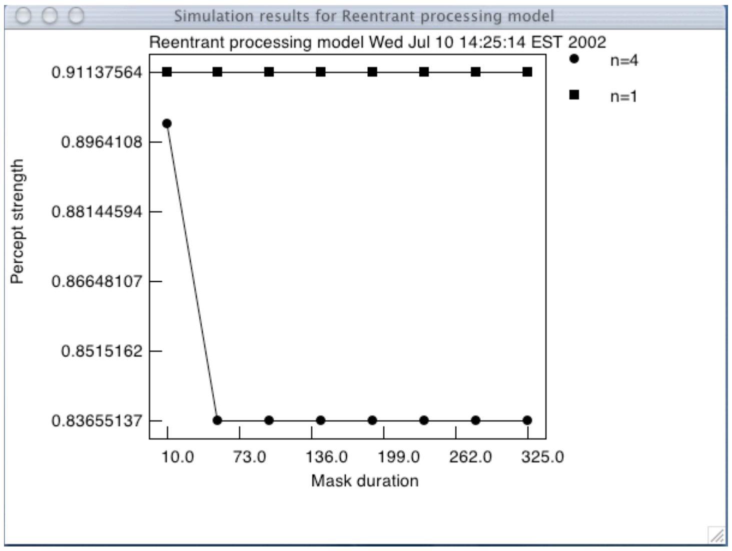
Start simulation

Figure 1:

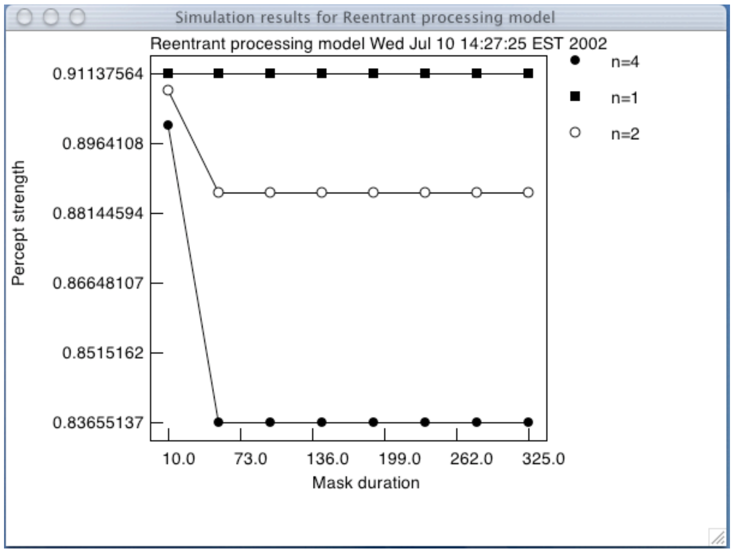
(a)



(b)



(c)



(d)

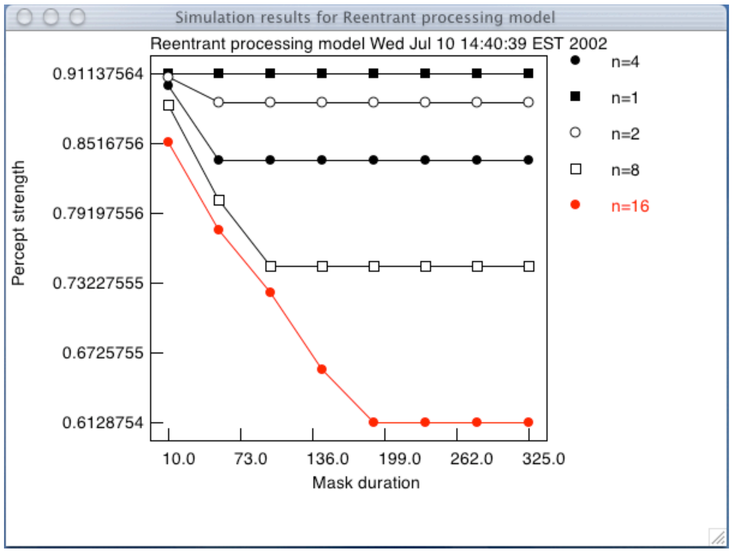
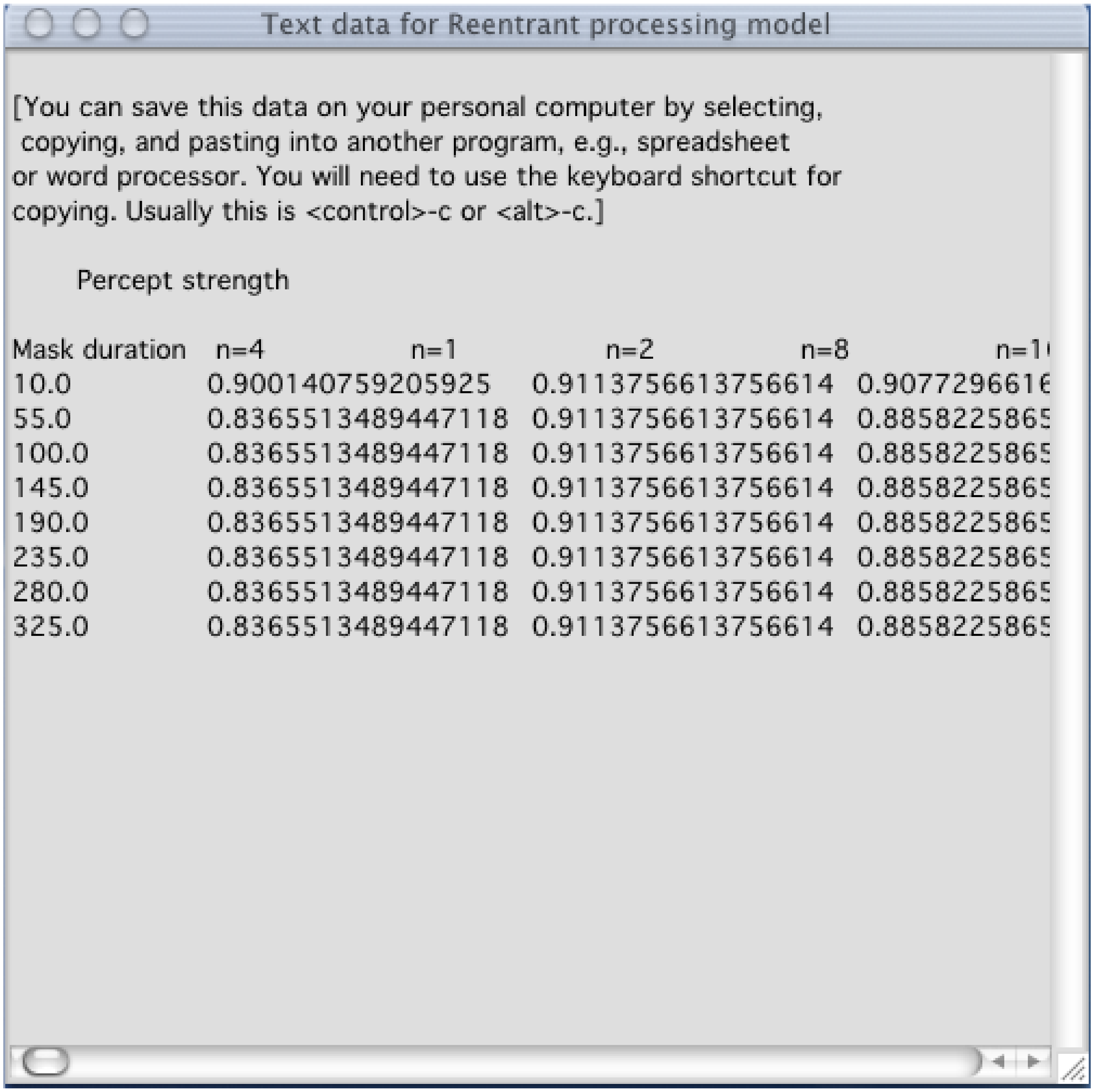


Figure 2:



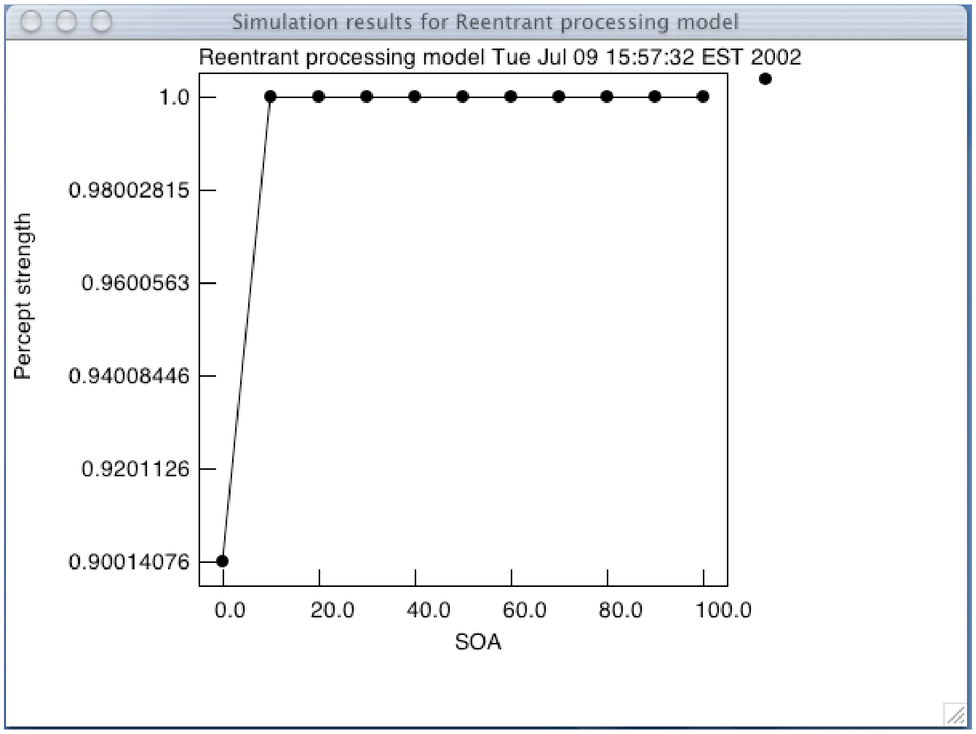
```
[You can save this data on your personal computer by selecting,
 copying, and pasting into another program, e.g., spreadsheet
or word processor. You will need to use the keyboard shortcut for
copying. Usually this is <control>-c or <alt>-c.]

      Percept strength

Mask duration   n=4              n=1              n=2              n=8              n=1
10.0            0.900140759205925   0.9113756613756614  0.9077296616
55.0            0.8365513489447118  0.9113756613756614  0.8858225865
100.0           0.8365513489447118  0.9113756613756614  0.8858225865
145.0           0.8365513489447118  0.9113756613756614  0.8858225865
190.0           0.8365513489447118  0.9113756613756614  0.8858225865
235.0           0.8365513489447118  0.9113756613756614  0.8858225865
280.0           0.8365513489447118  0.9113756613756614  0.8858225865
325.0           0.8365513489447118  0.9113756613756614  0.8858225865
```

Figure 3:

(a)


Simulation results for Reentrant processing model
Reentrant processing model Tue Jul 09 15:57:32 EST 2002
Percept strength
1.0
0.98002815
0.9600563
0.94008446
0.9201126
0.90014076
0.0
20.0
40.0
60.0
80.0
100.0
SOA

(b)

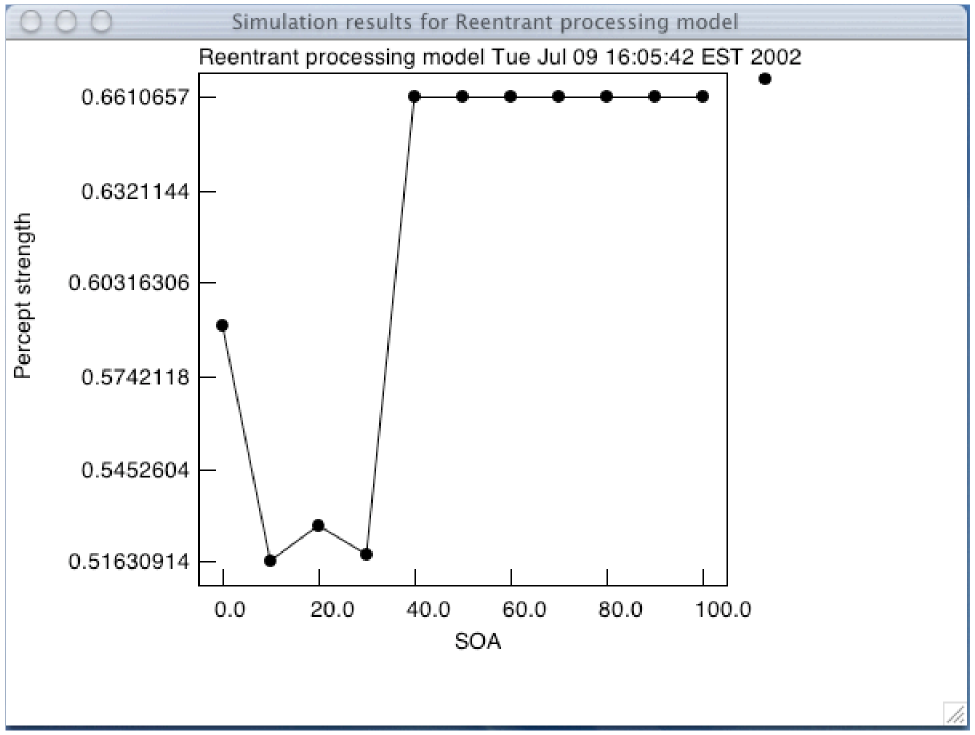

Simulation results for Reentrant processing model
Reentrant processing model Tue Jul 09 16:05:42 EST 2002
Percept strength
0.6610657
0.6321144
0.60316306
0.5742118
0.5452604
0.51630914
0.0
20.0
40.0
60.0
80.0
100.0
SOA

Figure 4:

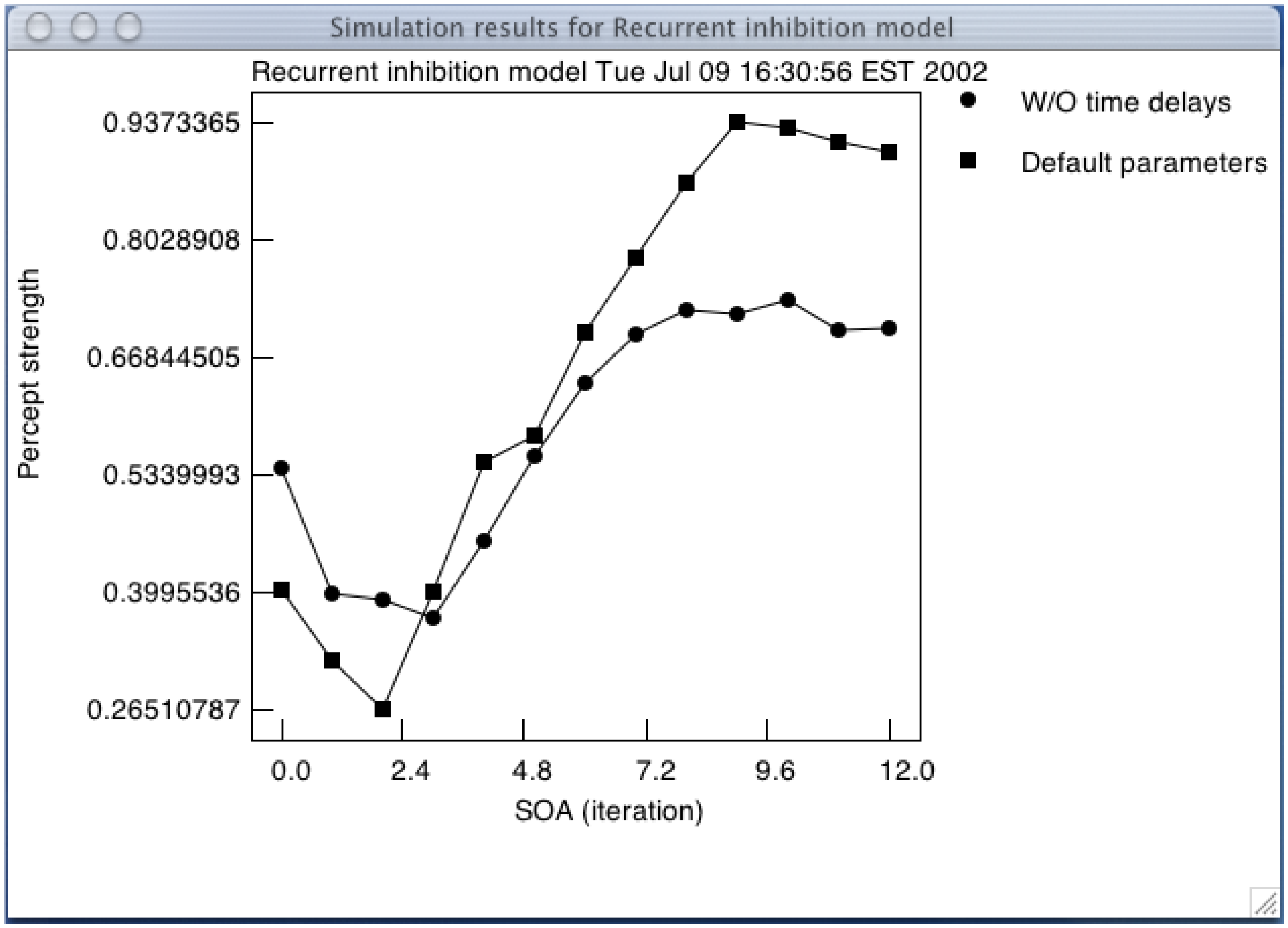
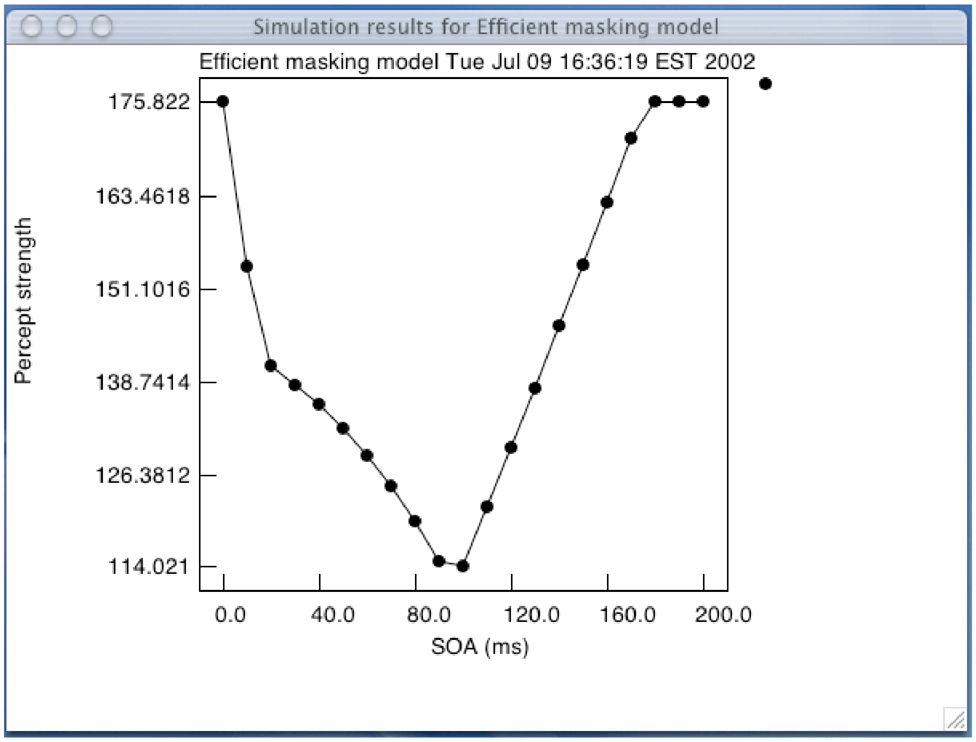

Simulation results for Recurrent inhibition model
Recurrent inhibition model Tue Jul 09 16:30:56 EST 2002
● W/O time delays
■ Default parameters
Percept strength
0.9373365
0.8028908
0.66844505
0.5339993
0.3995536
0.26510787
0.0
2.4
4.8
7.2
9.6
12.0
SOA (iteration)

Figure 5:

(a)


Simulation results for Efficient masking model
Efficient masking model Tue Jul 09 16:36:19 EST 2002
175.822
163.4618
151.1016
138.7414
126.3812
114.021
Percept strength
0.0
40.0
80.0
120.0
160.0
200.0
SOA (ms)

(b)

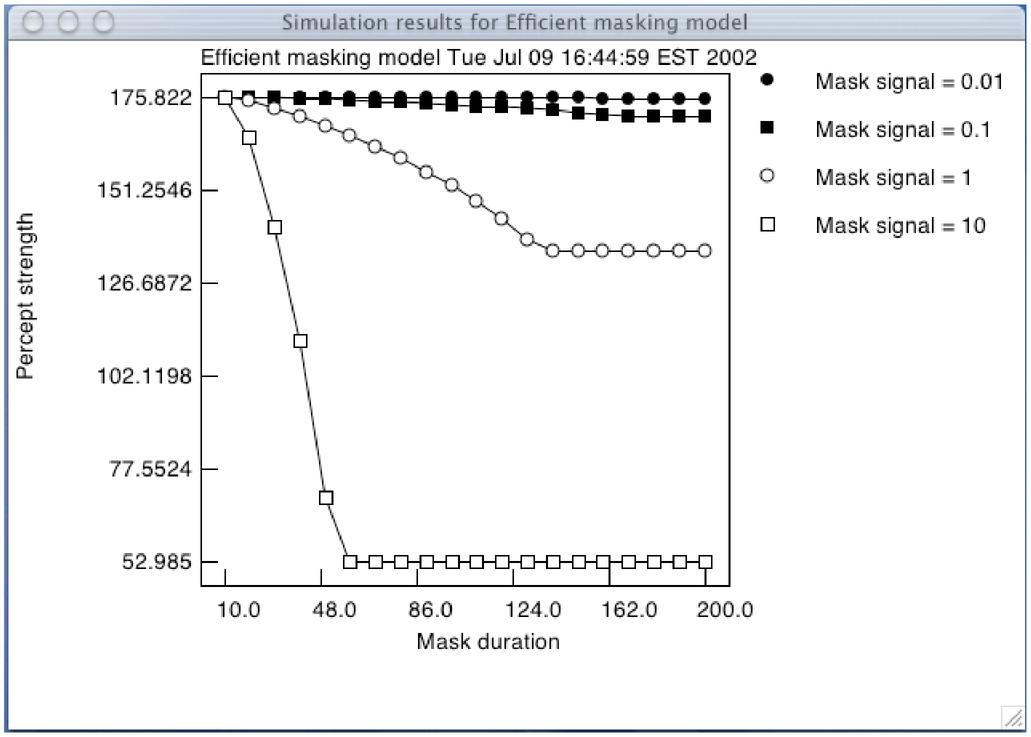

Simulation results for Efficient masking model
Efficient masking model Tue Jul 09 16:44:59 EST 2002
175.822
151.2546
126.6872
102.1198
77.5524
52.985
Percept strength
10.0
48.0
86.0
124.0
162.0
200.0
Mask duration
● Mask signal = 0.01
■ Mask signal = 0.1
○ Mask signal = 1
□ Mask signal = 10

Figure 6: