**Supplementary Information Guide**

**Data.mat**  Matlab data file containing a 100,000 row by 13 column array named Data1. The columns of Data1 are:  
1: Experimenter Code (1=Monika Krishan; 2 = Juliette Ye)  
2: Subject ID number (1-10)  
3: Session number (for a subject)  
4: Stimulus (0 = red, 1 = green)  
5: Subject's slider setting  
6: 6=Change Key (1 = clicked on to report a perceived change)  
7: 2nd Thought (1 = clicked on to report a 2nd thought); NaN for first 5 Subjects  
8: Reaction time in seconds (not analyzed)  
9: Observed *p*g, as determined by running the ideal free real-time observer  
 algorithm (true p)  
10: Objective change point; 1 = change in true (hidden) *p*g  
11: Flags changes in true *p* (not counting starts of sessions); sign indicates  
 direction of change  
12 = signed step heights (changes in slider position)

13 = step widths

**SubRws1.mat** Matlab data file containing a logical array with 10 columns; each column flags the observations from one subject. The logical vector in a single column may be used to extract from the Data1 array the data for a given subject.

**SesRws1.mat** Matlab data file containing a logical array with 10 columns; each column flags the data from a given session. ANDing a SubRws1 column vector and a SesRws1 column vector extracts from Data1 the data for a given session for a given subject

**Stimuli.mat** Matlab data file containing a 1,000 row by 100 column array. Each column contains the stimulus sequence for one of the 10 x 10 = 100 sessions (0 = red; 1 = green). This is the same information as in Column 4 of Data1, but laid out by session.

**ChangePointModel.m** Matlab script file. Download this function into Matlab's current directory along with the above files. Calling this function directly prompts the user to specify values for its parameters, then runs the model on the 100 sequences seen by the subjects. The syntax for the simplest direct call to this function is:

YourNameForOutput = ChangePointModel;  
  
When thus called, the function loads Data.mat, Stimuli.mat, SubRws1.mat and SesRws1.mat, then prompts the caller for the parameter values. The function returns a Matlab structure that contains the results from running the change-point model with the user-specified parameter values on each of the 100 stimulus sequences seen by our 10 subjects. A Notes field explains the contents of the other fields in the structure, to wit:

pHyper = initial values of the parameters of the betapdf prior on p\_g

pcHyper = initial values of the parameters of the betapdf prior on p\_c

BFcrit = decision criterion on posterior odds of a change in p\_g vs no change

KLcrit = decision criterion on nD\_KL(est\_p\_g||obs\_p\_g), the evidence for a problem  
 with p\_g

cp = 10x10 cell array. Each cell contains change points in model's ultimate

(retrospective) representation of the sequence of 1,000 stimuli seen

in one session by one subject. Rows are sessions; columns are Subjects

DP = Each cell contains the vector of the error-detection trials, the trials

at which the model detected a problem with the assumed value of p\_g

pc\_hat = final (session-end) estimates of p\_c

ps = the array of p values, one value for each of the segments delimited by

the change points in the cp field

Record = each cell contains a 10-col array recording the state of the

variables as of each of the detection trials in the DP field:

[alpha\_p beta\_p p\_hat alpha\_c beta\_c pc\_hat cp(end) DP(end) Nc Det];

ObsPs = Each cell contains the model's retrospective estimate of p\_g

for each of the 1,000 trials in a session--for comparison to its

real-time estimates and the subject's slider position. These are a

a stand-in for the Bayesian ideal observer's trial-by-trial p's

MdlPs = the model's real-time trial-by-trial estimates of pg

MdlStps = the trial between (the model's) successive slider steps

MdlStpHts = the heights of the model's steps (Delta p's)

MdlCPrprts = trials on which model called a change (model's real-time change calls)

CPhitRate = session-by-session hit rate in model's real-time change calls

CPflsAlarmRate = session-by-session false alarm rate in real-time change calls

CPhitLat = latencies btw true changes and model's call

The structure generated by a subsequent call may be adjoined to the structure generated by an earlier call, as follows:  
  
 YourNameForOutput(2) = ChangePointModel;

The code is extensively commented, which may help in understanding the model.

**IterateModelRuns.m**. Download and open this script to make repeated runs of the model varying its parameters from run to run. In the first cell of this script you specify hyperparameters on *p*g and *p*c, and sequen ces of values for *T*1 and *T*2, the decision criteria. After setting these values, call the script. It calls ChangePointModel repeatedly, feeding it different parameter values for *T*1 and*T*2 on each call, thereby producing a 2-layer Matlab structure, named Rslts\_T2. The values for *T*1, the criterion on *n*DKL,vary at the lower (2nd) level of Rslts; the values for *T*2, the criterion on the posterior odds of a change, vary at the first level. To access a result, specify indices at both levels, for example:  
  
 Rslts\_T2(2).T1(3).MdlCPrprts  
  
would show the model's change calls for the second of the user-specified values for *T*2 in combination with the third of the user-specified values for *T*1.

**ProbEstExperiment.m**

Download this function into Matlab's current directory in order to run the our version of Robinson’s experiment in Matlab. First screen prompts for subject’s initials, subject’s ID number and the session number for this subject. On completion of the session, the data are stored in a file named ‘CPexper\_Sub\_#\_#’, where the first # is the subject’s ID number and the 2nd is the session number. With a naïve subject, run ProbEstExperPractice function first, allowing them to become familiar with the task.

**ProbEstExperPractice.m** The practice version of the experiment. Run this first to read the instructions and practice performing the task, then run ProbEstExperiment. Those who do not have Matlab, can download stand-alone executable files (for PC or for Mac) at <http://cognitivegenetic.rutgers.edu/collars/>

**RunningAvModelFun.m** Download this function into Matlab’s current directory in order to run various versions of a 1- or 2-kernel running average model.

**IdealObserver\_c-code.tar.qz** The code that implements the ideal Bayesian observer computation of the “observed” *p*g’s