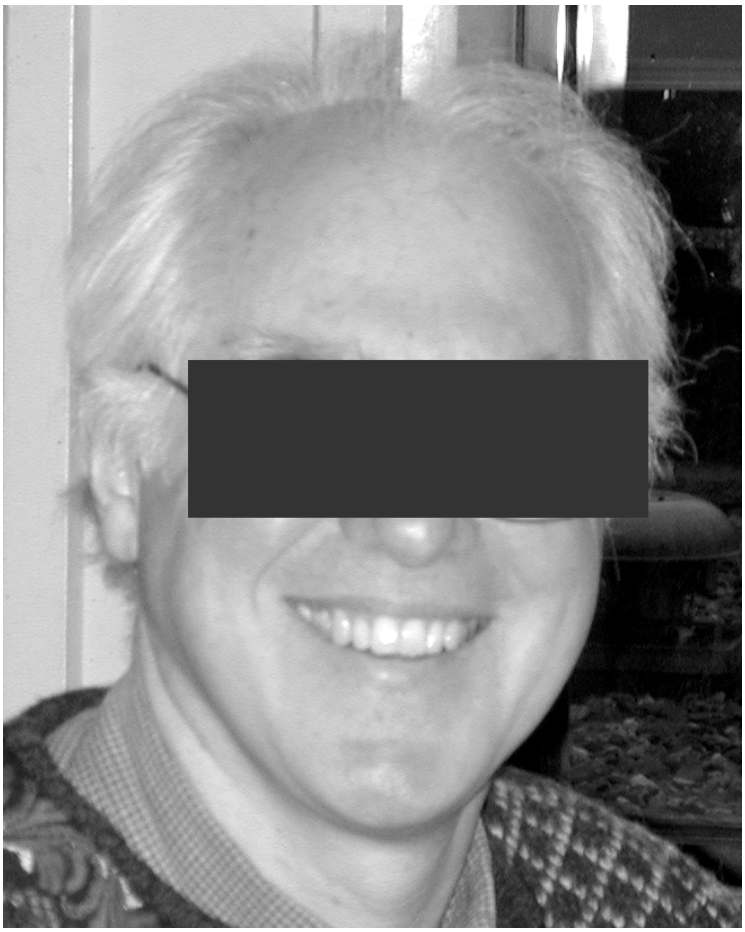
Psychology of Perception
Psychology 4165, Spring 2014
Laboratory 2

Face Recognition: Are the Eyes Important?



Intentionally Blank

Introduction

Recognition of human faces is a remarkably good. Even after 50 years have passed, people are able to choose which of two photographs is a high school classmate with an accuracy of almost 90 percent correct (Bahrick, Bahrick, & Wittlinger, 1975). The eyes and surrounding areas of the face are thought to be very important for successful face recognition (Bruce, 1988; Vinette, Gosselin, & Schyns, 2004). In this lab you test the hypothesis that obscuring the eyes makes it more difficult to later recognize a face.

There are two principle theories that describe how observers detect weak stimuli: The High Threshold Model (HTM) and Signal Detection Theory (SDT). In this lab you will also test the predictions of these two models for face recognition performance.

In the experiment you first will be shown a series of 64 photographs of faces to remember. Some will have their eyes covered and some will not. Then you will be tested with another series of 128 faces, half of which you have seen before and half of which are new. Your task is to decide on each trial whether or not the face had been previously seen. You will use a six-point confidence rating scale. A six-point rating scale corresponds to having five different decision criteria in the signal detection model. From your data you will calculate five hit rate – false alarm rate (HR–FAR) pairs (one pair for each of these response criteria) for the two types of faces (with eyes and without eyes. The two detection models will be compared to see how well each model can predict the observed data.

The objectives of this laboratory exercise are:

1. To test whether seeing the eyes enhance your ability to later recognize faces.
2. To test which model, the High Threshold Model or the Signal Detection Theory Model, better predicts your observed data in the face recognition experiment;

Experimental Procedure

You will first be shown a series of 64 target faces. Thirty-two of the faces have the eyes covered over and the other 32 are normal faces with the eyes visible. Study each face carefully during the 2 sec exposure time. Then in the test phase of the experiment, you will be shown a series of 128 test faces. Half of them will be target faces and half of them will be new faces that you have not previously seen. All the test faces will be in full

view (i.e., the eyes will not be covered). Using the six-point rating scale below rate each face on your confidence that it is a new face or a target face seen previously (with or without eyes):

1 = certain the face was not seen before
2 = perhaps the face was not seen before
3 = guessing that the face was not seen before
4 = guessing that the face was seen before
5 = perhaps the face was seen before
6 = certain the face was seen before

The experiment will be run on your computer using PsychoPy2, the free, state-of-the-art system for doing psychology experiment. Download the Lab2_Tools folder from the course web page:

http://psych.colorado.edu/~lharvey/P4165/P4165_2014_1_Spring/2014_Spring PSYC4165 Main Page.html

Open the Lab2_Tools folder and double click on the PsychoPy2 experiment file entitled *FaceRecognitionExperiment.psyexp* located in the Face-Eyes Experiment folder. If double-clicking does not start PsychoPy2, simply startup PsychoPy2 from the Applications folder, then drag the *FaceRecognitionExperiment.psyexp* file to the PsychoPy2 icon on the Dock at the bottom of the screen. To run the experiment, click on the large green Run button at the top of the window, the one with a runner in white on it. Enter your three initials in the dialog box. Don't worry about the difficulty of the task. Making these decisions sometimes is frustrating. Just relax and respond with a number from 1 to 6. Be sure, though, to use all of the response categories. Your data will be recorded automatically in a csv data file whose name starts with the initials you entered in the dialog box at the beginning of the experiment.

Data Analysis

1. Data Tabulation And Transformation

The raw data from this experiment are the number of times (frequencies) that you used each of the rating categories under each of the three stimulus conditions (new faces, old faces without eyes, and old faces with eyes). Your data analysis begins with a series of transformations of the raw data through the following sequence:

1. response scale frequencies;
2. response scale probabilities;

3. cumulative probabilities (hit rates and false alarm rates);
4. z-score transformation of the cumulative probabilities.

These transformations will be carried out by a special signal detection computer program, *RscorePlus*, but you first have to compute the frequencies of each rating scale number that you used. We have an R script for that!

First: Count up the number of times you used each of the rating categories for the new face trials, for the old faces no-eyes trials, and for the old faces with eyes trials. These are the rating scale frequencies. Record them in the column labeled "Rating Frequency" in Appendix II. This counting can be accomplished by the R script file *lab2_RawData.R*. Start up R with the script *lab2_RawData.R*. Edit line 11 that says "Typical_Student" and put your own name in its place. Execute all the commands in the script file. The script will ask you to select your raw data file. It is located in the Data folder in the Face-Eyes Experiment folder. Select the data file with the .csv extension. After you run the R script (*lab2_RawData.R*) the tabulated frequencies will be found in R objects **t.rec**. The script will also write the tabulated frequencies to a text file ("your name.txt") along with other required information that is needed by the *RscorePlus* program in the next step.

Second: Use the computer program *RscorePlus* to compute the remaining data transformations. *RscorePlus* will also fit a signal detection model to the data. Double click on *RscorePlus* to launch the program. Click on the "Open Data File" button and select the file produced by the R-script above ("your name.txt"). Select the model you want to fit to your data: Gaussian or High Threshold and click on the "Fit Model to Data" button. Select the other model and fit that model too. You may quite RscorePlus now. For each analysis, the program will produce four text files having the same name as your input file with these appended words:

1.	myfile_gauss_out.txt	a text file with the printed output of each analysis
2.	myfile_gauss_grf.txt	a text file to be imported into R for making graphs
3.	myfile_gauss_alt.txt	a results file for statistical analysis (not needed)
4.	myfile_gauss_vcv.txt	a variance-covariance matrix (not needed)

1.	myfile_htm_out.txt	a text file with the printed output of each analysis
2.	myfile_htm_grf.txt	a text file to be imported into R for making graphs
3.	myfile_htm_alt.txt	a results file for statistical analysis (not needed)

4. myfile_htm_vcv.txt a variance-covariance matrix (not needed)

If you want to print the *_gauss_out.txt and the *_htm_out.txt files open the files in Microsoft Word. Before printing, select all the text and set the font size to 8 and the font type to Courier or Monaco to make the printed results easier to read. Each file should then fit on both sides of a single piece of paper.

2. ROC Analysis

The high threshold model of detection predicts that the ROC will be a straight line when plotted in probability coordinates. The Gaussian signal detection theory model predicts that the ROC will be a straight line when plotted in quantiles (z-score) coordinates. The first step, therefore, is to plot your HR, FAR pairs on two types of graphs: one with linear probability coordinates and one with z-score coordinates. You may use the commands in file *lab2_Graphs.R* to plot these graphs. When you execute the *lab2_Graphs.R* script, you will be asked to choose an input file. Choose either the Gaussian grf file or the htm grf file. A plotting function, `plot1()` is defined by the script that will plot probability and quantile (z-score) ROC curves for the chosen model. Edit the script to put your own name on the graphs. Change "Your_Name" on line 17 to be your name. You might also need to change the position of the two labels for the ROC curves so they do not overlap the data points: Edit the x- and y- coordinates sent to the `text()` commands in the `plot1()` function.

RscorePlus computes the parameters of the best-fitting detection model using a maximum-likelihood technique called the method of scoring. The measure of detection sensitivity in the Gaussian detection model is d_a ("d-sub-a"), a generalized version of d' that is appropriate when the distributions do not have the same variance/standard deviation. In this experiment there are three signal (i.e, stimulus) conditions:

1. s0: new faces
2. s1: old faces first seen without eyes
3. s2: old faces first seen with eyes

For each pair of conditions there is a d_a. We are most interested in how eyes influence discrimination of old faces from new faces so we will focus on s0-s1 and s0-s2 comparisons. The d_a for discriminating the old faces seen without eyes from the new

faces (marked s0-s1 in the printout) and the d_a for discriminating the old faces seen with eyes from the new faces (marked s0-s2 in the printout) are available on the *_gauss_out.txt output file printout. Record these values and the goodness of fit (see below) information in the table in Appendix II on page 10.

For the high threshold model the measure of sensitivity is q, the probability that the signal exceeds the sensory threshold. The value of q for the without eyes condition (labeled oldNoE in the printout) and the value for the with-eyes condition (labeled oldEye in the printout) are given in the *_htm.out.txt file. Record these numbers and the goodness of fit (see below) information in Appendix II on page 11.

3. Graph the Signal Detection Model

RscorePlus computes the mean and standard deviation of the best-fitting Gaussian signal detection model from the data. It also computes the position of the five decision criteria. These data are printed in the *_gauss.out.txt and *_htm.out.txt files. You can draw the Gaussian SDT model, with criteria, using the `plot2()` command defined by executing the script file *lab2_Graphs.R*. When you run the script, it will ask you for the input file. Choose the *_gauss_grf.txt file. You should edit the script to put in your own name on the graphs (line 17). You may have to adjust the position of the text labels so they don't interfere with the plotted data. These labels are positioned with the `text()` command near the bottom of the R script. The first two arguments are the x- and y-coordinates of the text line.

4. Goodness of Fit

The ultimate test of each model is by how well it predicts the observed data. Since the parameters of each model were computed using a maximum-likelihood regression, the appropriate test is to calculate how well each model predicts the six response frequencies under the three stimulus conditions. The goodness of fit of these predictions can be formally computed using the chi square (χ^2) statistic. The value of χ^2 for each model is given on your printed output pages. Can you reject the null hypothesis for each model? If χ^2 is significantly larger than zero, a perfect fit, you should reject the model. Significance is evaluated by the p value associated with the χ^2. If it is less than, say,

0.01, reject the model. If you can reject a model, it should not be used for further analysis. Which is the better model: Gaussian SDT or HTM? You can't use χ^2 to compare the two models directly because the degrees of freedom for the High Threshold Model are different than for the Signal Detection Model because the High Threshold Model has fewer free parameters that must be computed from the data. To compare the fit of models having different numbers of free parameters we use a modified version of χ^2 that penalizes models with more free parameters. This modified χ^2 is called the Akaike Information Criterion (AIC) in honor of the Japanese statistician who developed it (Akaike, 1974). Which model fits your observed data better (i.e., has the lower AIC)? When you have finished your *RscorePlus* analysis, enter the appropriate results from the printed output on the summary sheet in Appendix II. Transfer these results to the group sheet that is at the front of the classroom. We will prepare a file with everyone's data for you to download, so that you can carry out further analyses next week.

5. Group Data

This section of the lab requires the group data file that will be available as soon as everyone in the class has analyzed their individual data. Go to the course web site and download the zip file "Lab 2 Group Data Files" and unzip (it should unzip automatically when you download; you might have to double-click on the file). You will test two null hypotheses:

1. H0: The signal detection model is just as good as the high threshold model;
2. H0: that seeing the eyes during the learning phase does not improve face recognition.

For the first hypothesis, compute a repeated measures analysis of variance on AIC, the modified *chi-square*, (the higher the AIC, the worse the fit). For the second hypotheses, compute a repeated measures analysis of variance on d_a, the index of sensitivity in the signal detection model. The script file *lab2_group_analysis.R* contains the commands for carrying out the analyses with R. Open the script in R and execute it. The script will ask for the input file. Locate the file `lab2_group_data_wide_2014_Spring.csv` and double click on it. The script defines two plotting functions for R. The first, `plot1()`

creates strip and box plots comparing the fit of the two models. The second, `plot2()` creates strip and box plots comparing the effect of eyes and no eyes on memory for faces as represented by d_a in the Gaussian signal detection model. The script also computes the group means and carries out an analysis of variance using a linear mixed effects model (`lme()` and/or `lmer()`)

6. Further reading

There is a large literature on signal detection theory. Although no additional reading is needed for this assignment, here are some references that you might wish to read if you want to enhance your understanding in this area. There are several books that are worthwhile (Egan, 1975; Green & Swets, 1966/1974; Macmillan & Creelman, 2005; McNicol, 1972; Swets & Pickett, 1982; Wickens, 2002). Here are some journal articles that will introduce you to the research literature (Harvey, 1992; Krantz, 1969; Simpson & Fitter, 1973; Swets, 1961, 1986a, 1986b; Swets, Tanner, & Birdsall, 1961).

Lab Report

Your lab report should contain six parts: Cover Page, Introduction, Methods, Results, Discussion, and References. In the introduction explain why you did the experiment. In the methods section describe what you did. In the results section present your findings, including graphs of your data. Your conclusions should be based on your statistical analyses, not your unsupported speculation. In the discussion you can let your creativity run wild. Give the reader your interpretation of the results. Discuss any implications and leads for further research. Laboratory reports must be typed, double-spaced on 8.5 x 11 paper with at least 1 inch margins. Conciseness and clarity are extremely important characteristics of good scientific writing. Strive for them. Worth 40 points. **Due in lab on 18 and 21 February 2014.**

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, AC-19*(6), 716–723.

Bahrick, H. P., Bahrick, P. O., & Wittlinger, R. P. (1975). Fifty years of memory for names and faces: A cross-sectional approach. *Journal of Experimental Psychology: General, 104*(1), 5475.

Bruce, V. (1988). *Recognising faces*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Egan, J. P. (1975). *Signal Detection Theory and ROC Analysis*. New York: Academic Press.

Green, D. M., & Swets, J. A. (1966/1974). *Signal detection theory and psychophysics* (A reprint, with corrections of the original 1966 ed.). Huntington, NY: Robert E. Krieger Publishing Co.

Harvey, L. O., Jr. (1992). The critical operating characteristic and the evaluation of expert judgment. *Organizational Behavior & Human Decision Processes, 53*(2), 229251.

Krantz, D. H. (1969). Threshold theories of signal detection. *Psychological Review, 76*(3), 308324.

Macmillan, N. A., & Creelman, C. D. (2005). *Detection theory: A user's guide* (2nd ed.). Mahwah, New Jersey: Lawrence Erlbaum Associates.

McNicol, D. (1972). *A primer of signal detection theory*. London: George Allen & Unwin.

Simpson, A. J., & Fitter, M. J. (1973). What is the best index of detectability? *Psychological Bulletin, 80*(6), 481488.

Swets, J. A. (1961). Is there a sensory threshold? *Science, 134*(3473), 168177.

Swets, J. A. (1986a). Form of empirical ROCs in discrimination and diagnostic tasks: Implications for theory and measurement of performance. *Psychological Bulletin, 99*(2), 181198.

Swets, J. A. (1986b). Indices of discrimination or diagnostic accuracy: Their ROCs and implied models. *Psychological Bulletin, 99*(1), 100117.

Swets, J. A., & Pickett, R. M. (1982). *Evaluation of diagnostic systems: methods from signal detection theory*. New York: Academic Press.

Swets, J. A., Tanner, W. P., Jr., & Birdsall, T. G. (1961). Decision processes in perception. *Psychological Review, 68*(5), 301340.

Vinette, C., Gosselin, F., & Schyns, P. G. (2004). Spatio-temporal dynamics of face recognition in a flash: It's in the eyes. *Cognitive Science, 28*(2), 289-301.

Wickens, T. D. (2002). *Elementary signal detection theory*. New York: Oxford University Press.

Appendix I: Summary of Raw Data from R Analysis of Data File

Rating Frequencies for the RscorePlus input file

	[1]	[2]	[3]	[4]	[5]	[6]	Total
New Faces							**64**
Old Faces no eyes							**32**
Old Faces with eyes							**32**

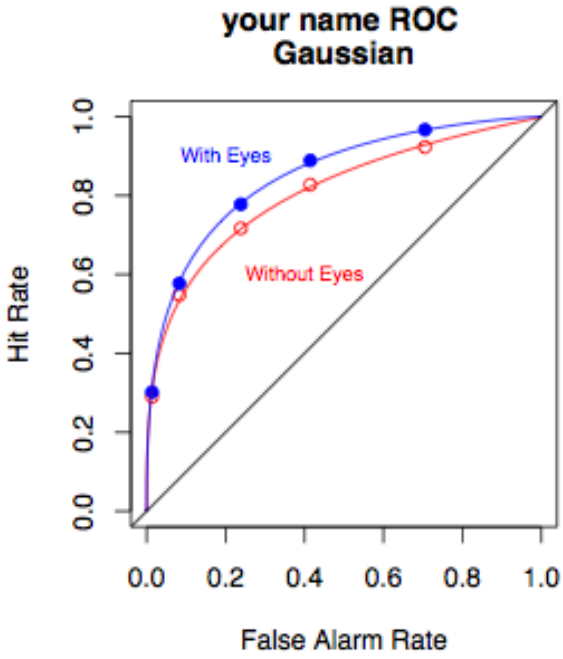
Appendix II: Summary of Results of Model-Fitting with RscorePlus
Signal Detection Theory Results

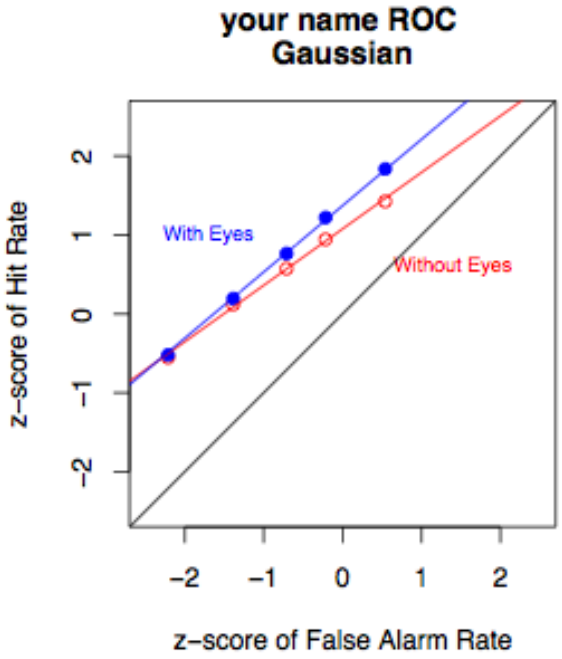
Sensory Process Sensitivity (d_a)		Goodness-of-Fit Measures		
Without eyes	With eyes	Chi-Square	Probability	AIC

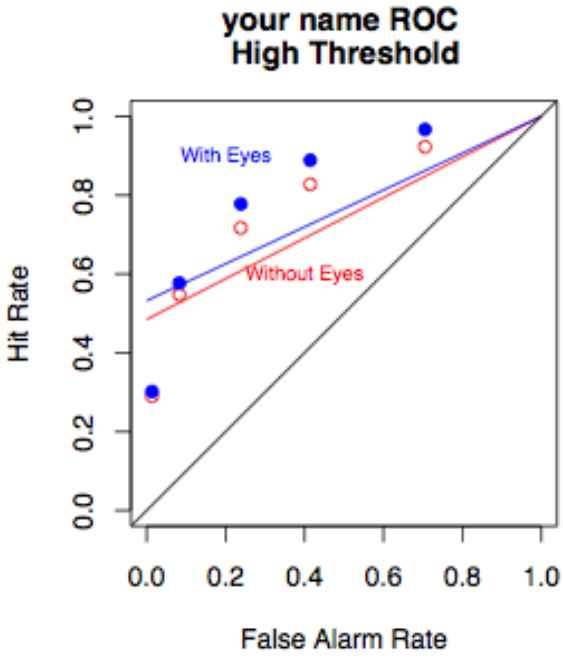
High Threshold Model Results

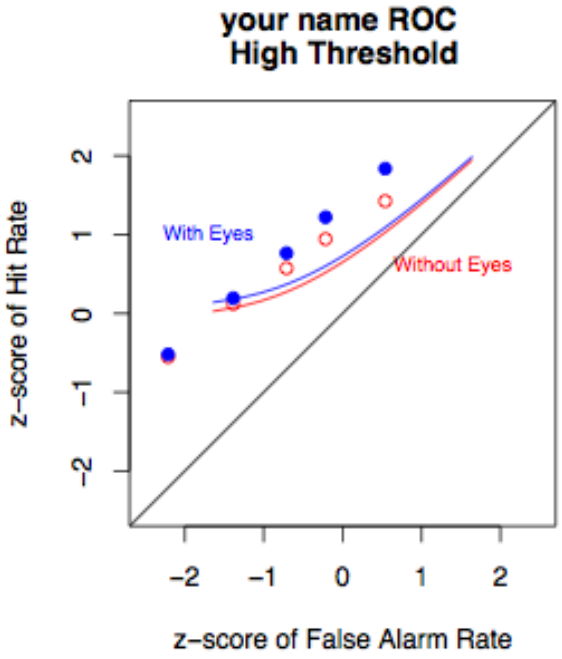
Sensory Process Sensitivity (q)		Goodness-of-Fit Measures		
Without eyes	With eyes	Chi-Square	Probability	AIC

Appendix III: Recommended format for the ROC graphs

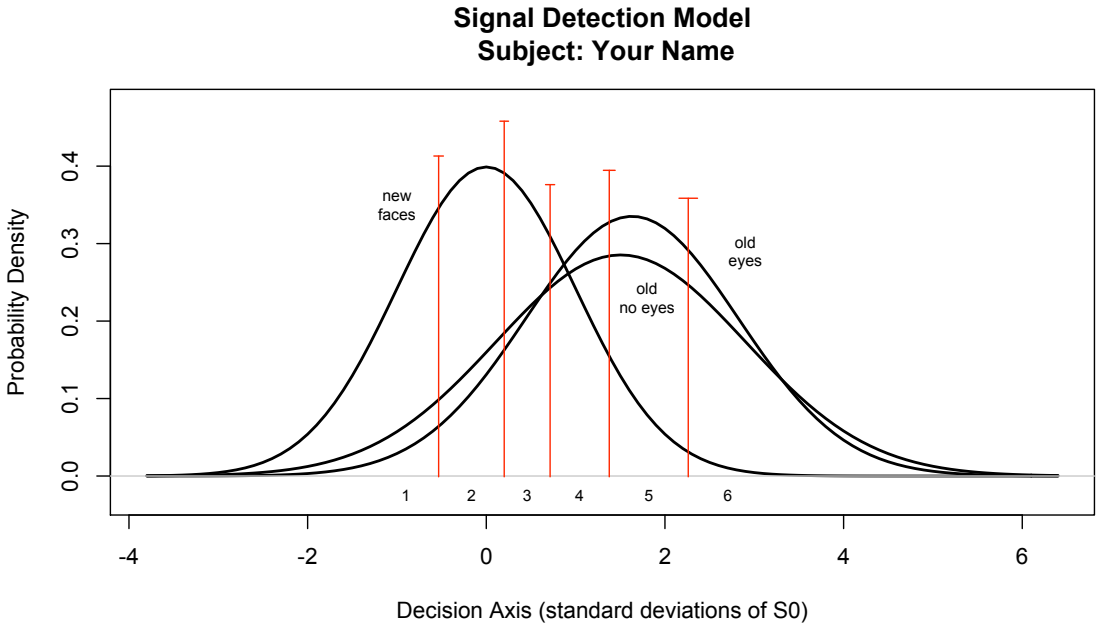








Appendix IV: Recommended format for the SDT model graph



Using R for Lab 2 Analysis

Analysis of raw PsychoPy2 data

```
# lab2_RawData.R
# 1 February 2014
# command file for reading in the PsychoPy2 csv data file to tabulate
# the response frequencies for use in RscorePlus.
# This script expects the input file to be the .csv file (opens in Excel)
# that was produced by the PsychoPy program that ran the experiment.
# The response frequency table is printed and a text output
# file is written that can be used by RscorePlus as input

# put your name here with no spaces. It will be used to create the output file
myName <- "Typical_Student"
fn <- file.choose()
df <- read.csv(fn, header = TRUE, nrows = 64+128)
# make data frame with just the rating trials data
df.rec <- df[ 65:192, ]
# add new column called "Signal" for the SDT analysis
df.rec <- cbind(df.rec, Signal = df.rec$Eyes)
# add a new level to the factor "Signal" and call it "new"
df.rec[, "Signal"] <- factor(df.rec[, "Signal"], levels=c("new","noeyes","eyes"))
# set all the new face signal trials to "new"
df.rec[df.rec$Target=="new", "Signal"] <- "new"
# compute frequency contingency table and print it
t <- with(df.rec, table(Signal, response.keys))
print(t)
# write out a text file for RscorePlus input
tab <- "\t"       # tab character
eol <- "\r"       # Mac end-of-line character
# open the output file
outputFileName <- paste(myName,".txt", sep = "")
of <- file(outputFileName, open = "wt")
# write the first line to file
cat("Heading", eol, sep = "", file = of)
# create text lines for output file
l1 <- paste(myName, eol, sep = "")
l2 <- paste("6",tab,"3",tab,"1",tab,"0",tab,"0",tab,"1",tab,"SINT",eol,sep="")
l3 <-
 paste("labels",tab,"R_1",tab,"R_2",tab,"R_3",tab,"R_4",tab,"R_5",tab,"R_6",eol,sep="")
l4 <- paste("new",
 tab,t[1,1],tab,t[1,2],tab,t[1,3],tab,t[1,4],tab,t[1,5],tab,t[1,6],eol,sep="")
l5 <-
 paste("oldNoE",tab,t[2,1],tab,t[2,2],tab,t[2,3],tab,t[2,4],tab,t[2,5],tab,t[2,6],eol,sep=
 "")
l6 <-
 paste("oldEye",tab,t[3,1],tab,t[3,2],tab,t[3,3],tab,t[3,4],tab,t[3,5],tab,t[3,6],eol,sep=
 "")
l7 <- paste("0.0",tab,"1.0",tab,"0.5",tab,"1.0",tab,"1.0",tab,"1.0",eol,sep="")
l8 <- paste("0",tab,"0",tab,"1",tab,"1",tab,"1",tab,"1",eol,sep="")
# write lines to the output file
cat(l1, sep = "", file = of)
cat(l2, sep = "", file = of)
cat(l3, sep = "", file = of)
cat(l4, sep = "", file = of)
cat(l5, sep = "", file = of)
cat(l6, sep = "", file = of)
cat(l7, sep = "", file = of)
cat(l8, sep = "", file = of)
# write lines at the end of the file
cat("end of file", eol, sep = "", file = of)
cat("-1", eol, sep = "", file = of)
close(of)
print(paste("The frequencies have been written to file ", outputFileName))
print("Use it as input to RscorePlus")
```

Drawing Graphs from RscorePlus Output

```
# lab2_SDT_Graphs.R
#
# R script file for defining two plotting
# functions for making signal detection
# plots from the data in the *_grf file
# produced by RscorePlus:
#
# plot1() plots a probability ROC and
#     a z-score ROC in a single figure;
#
# plot2() plots the best-fitting Gaussian
#     signal detection model.
#
# 1 February 2014

# Edit below to put in your own name
myName = "Your_Name"

# read the RscorePlus graphic file (*_grf) into a data frame
fn <- file.choose()                              # choose the file name
df <- read.delim(fn, header=TRUE, check.names=TRUE)

# check that the input file is an RscorePlus graphics file
doGrf <- as.logical(grepl("_grf", fn))
if (doGrf) {
 md = paste(df$Model[1], "Model")  # name of model

 plot1 <- function() {
      # plot ROC curve for without eyes in red and with eyes in blue
      par(mfrow = c(1,2))     # two graphs per figure
      par(pty = "s")          # make plots square

      # probability plot
      plot(Obs_pYs1 ~ Obs_pYs0,
            data = df,
            pch = 21, col = "red",
            xlim = c(0, 1),
            ylim = c(0, 1),
            xlab = "False Alarm Rate",
            ylab = "Hit Rate",
            main = paste(myName, md, "Probability ROC", sep="\n"))
      points(Obs_pYs2 ~ Obs_pYs0, data=df, pch=19, col="blue")
      lines(Pred_pYs1 ~ Pred_pYs0, data=df, col="red", lwd = 2)
      lines(Pred_pYs2 ~ Pred_pYs0, data=df, col="blue", lwd = 2)
      text(.4, .60, "Without\nEyes", col="red", cex=0.75)
      text(.2, .90, "With\nEyes", col="blue", cex=0.75)
      abline(0,1)

      # z-score plot
      plot(Obs_zYs1 ~ Obs_zYs0, data=df, pch=21, col="red",
            xlim=c(-2.5, 2.5),
            ylim=c(-2.5, 2.5),
            xlab="z-score of False Alarm Rate",
            ylab="z-score of Hit Rate",
            main = paste(myName, md, "Z-Score ROC", sep="\n"))
      points(Obs_zYs2 ~ Obs_zYs0, data=df, pch=19, col="blue")
      lines(Pred_zYs1 ~ Pred_zYs0, data=df, col="red", lwd=2)
      lines(Pred_zYs2 ~ Pred_zYs0, data=df, col="blue", lwd=2)
      text(1.4, .6, "Without\nEyes", col="red", cex=0.75)
      text(-1.7, 1, "With\nEyes", col="blue", cex=0.75)
      abline(0,1)

      par(pty = "m")          # restore the normal size of plots
      par(mfrow = c(1,1))     # restore the single plot per window mode
 } # end of plot1()

 plot2 <- function() {
      mdName = paste(df$Model[1], "Detection Model")
```

```
      # check that the model is Gaussian
      doPlot <- pmatch("Gaussian", df$Model, nomatch=FALSE)

      if(doPlot) {
            # plot distribution for without eyes in red
            # and with eyes in blue

            # ########################################################
            # Figure out how many criteria and how many signal conditions
            nsig <- df$nsig[1]      # number of signals
            nrat <- df$nrat[1]      # number of ratings
            ncrt <- nrat - 1  # number of criteria

            # Plot the signal detection model
            # find the maximum density value for setting y-axis
            # find the s0 probability density column: save in ind0
            # find the last signal column: save in ind1
            ind0 <- match("Pred_Density0", names(df))
            ind1 <- ind0 + nsig - 1
            max.density <- 1.3 * max(df[, ind0:ind1], na.rm = TRUE)
            par(pty = "m")
            # Set up the plot
            plot(df[, ind0] ~ Xc, data = df, type = "n",
                  lwd = 2, col = "black",
                  xlim = c(min(df$Xc, na.rm = TRUE), max(df$Xc, na.rm = TRUE)),
                  ylim = c(-0.03, max.density),
                  xlab = "Decision Axis (standard deviations of S0)",
                  ylab = "Probability Density",
                  main = paste(myName, mdName, sep = "\n"))
            # plot the probability density functions (s0, s1 ...)
            colors <- c("black", "red", "blue")
            for (i in 0:(nsig-1)){
                  lines(df[, ind0+i] ~ Xc, data = df,
                  type = "l",
                  lwd = 2,
                  col = colors[i+1])
            }
            abline(h = 0, col = "gray", lty = "solid")
            # extract the criterion coordinates
            crit <- subset(df, Criteria != is.na(Criteria))[c("Xc", "Xc_se", "Criteria")]
            xc <- df$Xc[1:ncrt]
            # plot the decision criterion markers
            for (i in 1:ncrt){
                  segments(crit$Xc[(2*i)-1], crit$Criteria[(2*i)-1],
                        crit$Xc[2*i], crit$Criteria[2*i], col="darkgreen")}
            # plot the standard error of each criterion marker
            for (i in 1:ncrt){
                  segments(crit$Xc[2*i]-crit$Xc_se[2*i], crit$Criteria[2*i],
                  crit$Xc[2*i]+crit$Xc_se[2*i], crit$Criteria[2*i], col = "darkgreen")}
            # label the response categories
            xc.lab <- rep(0, nrat)
            xc.lab[2:ncrt] <- xc[1:ncrt-1] + diff(xc)/2
            xc.lab[1] <- xc[1] - (diff(xc)/2)[1]
            xc.lab[nrat] <- xc[ncrt] + (diff(xc)/2)[ncrt-1]
            text(xc.lab, -0.025, labels = 1:nrat, cex = 0.7, col="darkgreen")

            # label the distributions
            text(-1, .35, "new\nfaces", cex=0.7, col=colors[1])
            text(1.8, .23, "old\nno eyes", cex=0.7, col=colors[2])
            text(2.9, .29, "old\neyes", cex=0.7, col=colors[3])
      } else
      {
            print("Function plot2() can only plot Gaussian models.")
            print("Select the Gaussian grf file as input.")
      } # end of doPlot if statement
 } # end of plot2()
} else {
 print("The input file must be a grf file from RscorePlus.")
} # end of doGrf if file test at top
```

Computing ANOVAs on Group Data using R

```
# lab2_group_analysis.R
# PSYCH 4165 Lab 2
# Command file for using lme() to compute a
# repeated measures analysis of variance in R.
# Lew Harvey
# 9 February 2014

# if you want to use lmer() instead of lme(), set this flag to TRUE
USE_LME4 = FALSE
if (USE_LME4) {
 require(lme4)          # load the nlme package to use lme()
} else {
 require(nlme)          # load the lme4 package to use lmer()
}

# Read in group data in wide format from text file:
# fn <- file("lab2_group_data_wide_2014_Spring.csv")
fn <- file.choose()
df.wide <- read.csv(fn, header=TRUE)

# ****************************************************
# reshape the file into long format
# Note: reshape() is a bit of a pain to use but what it does
# is take data that are in a so-called wide format
# (one row per subject, separate columns per dependent variable)
# and put it into the long format needed by many R analysis routines
# (multiple rows per subject, single column per dependent variable)
df.long <- reshape(df.wide,
 varying = list(c("da_no_eyes", "da_with_eyes"), c("htm_no_eyes", "htm_with_eyes"),
 c("sdt_AIC", "htm_AIC")),
 v.names = c("da", "htm_p", "aic"),
 timevar = c("eyes", "mod"),
 idvar = "subj",
 times = c("noEyes", "withEyes"),
 direction = "long")

rownames(df.long) <- 1:dim(df.long)[1]
df.long[df.long$mod == "noEyes", "mod"] <- "sdt"
df.long[df.long$mod == "withEyes", "mod"] <- "htm"
df.long$eyes <- factor(df.long$eyes)
df.long$mod <- factor(df.long$mod)
df.long$section <- factor(df.long$section)
df <- df.long

# for drawing lines between pairs of data
df.sdt      <- subset(df, mod=="sdt")
df.htm      <- subset(df, mod=="htm")
df.withEyes <- subset(df, eyes=="withEyes")
df.noEyes   <- subset(df, eyes=="noEyes")

# ****************************************************
# make functions to draw strip and box plots
# plot1()  plots AIC fit against the two models
# plot2()  plots memory sensitivity for eyes/no eyes
# ****************************************************

# Write out a summary of the variables in the data frame:
print(summary(df))

# ****************************************************
# plotting section
# define four functions to plot the two sets of graphs
# plot1()
# plot1a()
# plot2()
# plot2a()
# To make the plots, type the command
# ****************************************************
```

```
plot1 <- function() {
 # set graphic parameters to plot two graphs in one panel
 par(mfcol=c(1,2))
 # Make a strip chart of the data
 stripchart(aic ~  mod, data = df,
      vertical = TRUE,
      method = "jitter",  jitter = 0.03,
      xlim = c(0.75, 2.25),
      xlab = "Detection Model",
      ylab = "Badness-of-Fit (AIC)",
      main = "Lab 2 Group Data")

 # Make a box plot of the data:
 boxplot(aic ~ mod, data = df,
      col = "yellow",
      horizontal = FALSE,
      xlab = "Detection Model",
      ylab = "Badness-of-Fit (AIC)",
      main = "Lab 2 Group Data")

 par(mfcol=c(1,1))
}

# plot1a draws the same plot as plot1() but adds
# lines connecting data points from the same subject
plot1a <- function() {
 par(mfcol=c(1,2))      # two plots per figure
 # Make a strip chart of the data
 stripchart(aic ~  mod, data = df,
      vertical = TRUE,
      method = "jitter",  jitter = 0.03,
      xlim = c(0.75, 2.25),
      xlab = "Detection Model",
      ylab = "Badness-of-Fit (AIC)",
      main = "Lab 2 Group Data")

 # connect corresponding points with a line
 segments(x0 = rep(1, length(df.htm$aic)), y0 = df.htm$aic,
             x1 = rep(2, length(df.sdt$aic)), y1 = df.sdt$aic)

 # Make a box plot of the data:
 boxplot(aic ~ mod, data = df,
      col = "yellow",
      horizontal = FALSE,
      xlab = "Detection Model",
      ylab = "Badness-of-Fit (AIC)",
      main = "Lab 2 Group Data")

 par(mfcol=c(1,1))
}


# Test the hypothesis that eyes have no influence on face memory
# Memory sensitivity is measured with d-sub-a, the generalized
# signal detection theory measure d-prime
plot2 <- function() {
 par(mfcol=c(1,2))      # two plots per figure
 # Make a strip chart of the data
 stripchart(da ~ eyes, data = df, vertical = TRUE,
      method = "jitter", jitter = 0.03,
      xlim = c(0.75, 2.25),
      xlab = "Eyes (Yes or No)",
      ylab = "Face Memory Sensitivity (Da)",
      main = "Lab 2 Group Data")

 # Make a box plot of the data
 boxplot(da ~ eyes, data = df, horizontal = FALSE,
      col = "yellow",
      xlab = "Eyes (Yes or No)",
      ylab = "Face Memory Sensitivity (Da)",
```

```
      main = "Lab 2 Group Data")

 par(mfcol=c(1,1))      # two plots per figure
}

plot2a <- function() {
 par(mfcol=c(1,2))      # two plots per figure
 # Make a strip chart of the data
 stripchart(da ~ eyes, data = df, vertical = TRUE,
      method = "jitter", jitter = 0.03,
      xlim = c(0.75, 2.25),
      xlab = "Eyes (Yes or No)",
      ylab = "Face Memory Sensitivity (Da)",
      main = "Lab 2 Group Data")

 # connect corresponding points with a line
 segments(x0 = rep(1, length(df.noEyes$da)),   y0 = df.noEyes$da,
             x1 = rep(2, length(df.withEyes$da)), y1 = df.withEyes$da)

 # Make a box plot of the data
 boxplot(da ~ eyes, data = df, horizontal = FALSE,
      col = "yellow",
      xlab = "Eyes (Yes or No)",
      ylab = "Face Memory Sensitivity (Da)",
      main = "Lab 2 Group Data")

 par(mfcol=c(1,1))      # one plot per figure
}

# test the hypothesis that viewing the eyes does not influence memory
# Compute the repeated measures ANOVA to test the null hypothesis
# that having eyes or no-eyes does not influence memory sensitivity (d-sub-a)

if (USE_LME4) {
 print("", quote=FALSE)
 print("d-sub-a linear mixed effects model (lmer)")
 lmer.mod.da <- lmer(da ~ eyes + (1 | subj), data = df)
 print(anova(lmer.mod.da))
} else {
 print("", quote=FALSE)
 print("d-sub-a linear mixed effects model (lme)")
 lme.mod.da <- lme(da ~ eyes, random = ~ 1 | subj, data = df)
 print(anova(lme.mod.da))
}

# print group means
print("", quote=FALSE)
print("d-sub-a Means")
with(df, print(cbind(mean = tapply(da, eyes, mean),
 std.dev = tapply(da, eyes, sd),
 n = tapply(da, eyes, length))))



# test the hypothesis that the two models
# fit the data equally well using AIC as the
# badness of fit measure: the bigger the AIC,
# the worse the fit of the model.
# Compute a repeated measures ANOVA

if (USE_LME4) {
 print("", quote = FALSE)
 print("Badness-of-Fit linear mixed effects model (lmer)")
 lmer.mod.aic <- lmer(aic ~ mod + (1 | subj), data = df)
 print(anova(lmer.mod.aic))
} else {
 print("", quote = FALSE)
 print("Badness-of-Fit linear mixed effects model (lme)")
 lme.mod.aic <- lme(aic ~ mod, random = ~ 1 | subj, data = df)
 print(anova(lme.mod.aic))
}
```

```
# print group means
with(df, print(cbind(mean = tapply(aic, mod, mean),
 std.dev = tapply(aic, mod, sd),
 n = tapply(aic, mod, length))))


plot1()
```