Psychology 205: Research Methods in Psychology
Using R to examine our data: Part II

William Revelle

Department of Psychology
Northwestern University
Evanston, Illinois USA

April, 2021

Outline

Our memory study

1. The memory study had 2 parts.
2. It was inspired and partly based upon the work of Roediger & McDermott (1995)
3. Shortened to make it suitable for a class experiment.
4. Eight lists of 15 words were presented.
5. Following each list, two minutes were given for recall
6. Recall of the 15 words for each of 8 lists
7. After all lists had been presented, a recognition task was given for a subset of the words.
    - If our data match prior verbal learning data, there should be a serial position of the probability of recall.
    - First and last words from the list should be more frequently recalled.
8. Recognition of the words presented (and not presented)
    - Recognition of 4 words per list (3 presented, one not presented)
    - Recognition of 16 words (8 cued by the lists, 8 not cued) (We reported this part on Monday)

The word lists

Table: The first 8 word lists from Roediger & McDermott (1995)

Position	L1	L2	L3	L4	L5	L6	L7	L8
CUE	anger	black	bread	chair	cold	doctor	foot	fruit
1	mad	white	butter	table	hot	nurse	shoe	apple
2	fear	dark	food	sit	snow	sick	hand	vegetable
3	hate	cat	eat	legs	warm	lawyer	toe	orange
4	rage	charred	sandwich	seat	winter	medicine	kick	kiwi
5	temper	night	rye	couch	ice	health	sandals	citrus
6	fury	funeral	jam	desk	wet	hospital	soccer	ripe
7	ire	color	milk	recliner	frigid	dentist	yard	pear
8	wrath	grief	flour	sofa	chilly	physician	walk	banana
9	happy	blue	jelly	wood	heat	ill	ankle	berry
10	fight	death	dough	cushion	weather	patient	arm	cherry
11	hatred	ink	crust	swivel	freeze	office	boot	basket
12	mean	bottom	slice	stool	air	stethoscope	inch	juice
13	calm	coal	wine	sitting	shiver	surgeon	sock	salad
14	emotion	brown	loaf	rocking	Arctic	clinic	smell	bowl
15	enrage	gray	toast	bench	frost	cure	mouth	cocktail

Notice that the words are in descending strength of association with the (unpresented) cue word.

Using R to analyze our data – Some preliminaries

Note that the R code is to show how I did it, you do not need to do this (unless you want). We will do subsequent analyses using R so it is useful to try to *understand* what I am doing. But you can just take the results I give for your paper.

1. Ideally, you have already done these preliminary steps!
2. Make sure that you have installed R on your computer.
3. Make sure that you then installed the *psych*.
4. See the short and longer tutorials on how to do this.
5. Once they are installed (you only need to do this once) then you need to use the `library(psych)` command at the beginning of every session.

Using R to examine the experimental data

The data from our experiment were taken from the Qualtrics file and saved as two .csv files from Excel. The web address is http://personality-project.org/courses/205/205.recognition.21.csv and http://personality-project.org/courses/205/205.recall.csv. You can view them in your browser or read it directly into R. If you read it in your browser, you can select it all, copy the data to your clipboard, and read the clipboard (option 1)

R code

```
library(psych)  #start each session with this
#go to the website to read the data for today
#Option 1  Copy the file to your clipboard and then read the clipboard
recall  <- read.clipboard.csv()
#option 2  Read the remote file directly
filename <-
   "http://personality-project.org/courses/205/205.recall.csv"
recall <- read.file(filename)  #read the data
dim(recall)  #find the dimensions of the data (subjects x variables)
```

```
filename <-
+    "http://personality-project.org/courses/205/205.recall.csv"
Data from the .csv file http://personality-project.org/courses/205/205.recall.csv has been loaded.
> dim(recall)  #find the dimensions of the data (subjects x variables)
```

Make sure we have the data

Always describe the data you are about to analyze. We use the describe function. But this will produce 120 lines (one for each word). We can get a glimpse of this summary by saving the output and then just looking at the first and last 8 lines.

R code

```
d <- describe(recall)  #this will be 120 rows long
headTail(d)  #just show a few items
```

```
 d <- describe(recall)
headTail(d,top=8,bottom=8)
         vars  n mean   sd median trimmed  mad min max range  skew kurtosis   se
mad         1 14 0.71 0.47      1    0.75    0   0   1     1 -0.85    -1.36 0.13
fear        2 14 0.64  0.5      1    0.67    0   0   1     1 -0.53    -1.83 0.13
hate        3 14 0.43 0.51      0    0.42    0   0   1     1  0.26    -2.07 0.14
rage        4 14 0.71 0.47      1    0.75    0   0   1     1 -0.85    -1.36 0.13
temper      5 14 0.57 0.51      1    0.58    0   0   1     1 -0.26    -2.07 0.14
fury        6 14 0.43 0.51      0    0.42    0   0   1     1  0.26    -2.07 0.14
ire         7 14 0.64  0.5      1    0.67    0   0   1     1 -0.53    -1.83 0.13
wrath       8 14 0.64  0.5      1    0.67    0   0   1     1 -0.53    -1.83 0.13
...       ...  ...  ...  ...    ...     ...  ... ... ...   ...   ...      ...  ...
banana    113 14 0.86 0.36      1    0.92    0   0   1     1 -1.83     1.45  0.1
berry     114 14 0.43 0.51      0    0.42    0   0   1     1  0.26    -2.07 0.14
cherry    115 14 0.64  0.5      1    0.67    0   0   1     1 -0.53    -1.83 0.13
basket    116 14  0.5 0.52    0.5     0.5 0.74   0   1     1     0    -2.14 0.14
juice     117 14 0.86 0.36      1    0.92    0   0   1     1 -1.83     1.45  0.1
salad     118 14 0.71 0.47      1    0.75    0   0   1     1 -0.85    -1.36 0.13
bowl      119 14 0.86 0.36      1    0.92    0   0   1     1 -1.83     1.45  0.1
cocktail  120 14    1    0      1       1    0   1   1     0   NaN      NaN    0
```

Combine lists to examine serial position effects

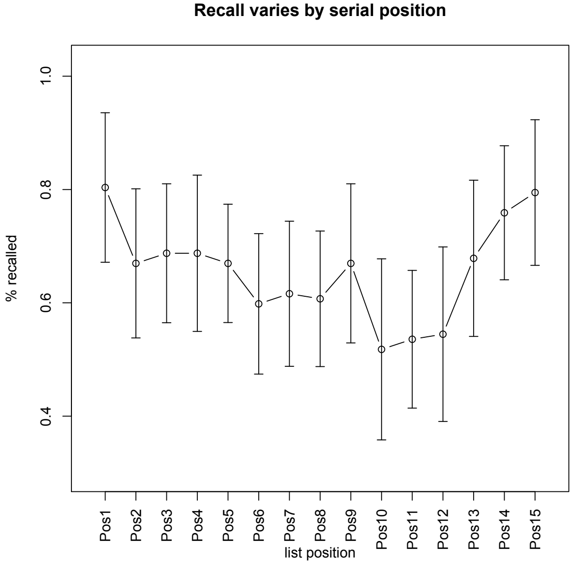
This is a bit of "R" magic

R code

```
n.items <- 15  #number of items per list
items <- seq(1,120, by=n.items ) #where is the first item for each list
items # show what we did
position.list <- list()  #get ready to store the results somewhere
for(i in 1:n.items ) {position.list[[i]] <- rowSums(recall[items + i-1])} #find the position information
position.list <- matrix(unlist(position.list),ncol=15)
colnames(position.list) <- paste0("Pos",1:NCOL(position.list))
rownames(position.list) <- paste0("S",1:NROW(position.list))
position.list   #see what we produced
error.bars(position.list/8,ylab="% recalled",xlab="list position",
       main="Recall varies by serial position",typ="b", eyes=FALSE,las=2)
```

```
 items
[1]   1  16  31  46  61  76  91 106
postion.lists
    Pos1 Pos2 Pos3 Pos4 Pos5 Pos6 Pos7 Pos8 Pos9 Pos10 Pos11 Pos12 Pos13 Pos14 Pos15
S1     8    8    7    7    5    7    5    6    5     6     7     6     7     5     7
S2     6    4    4    7    7    5    3    4    5     2     3     3     7     5     8
S3     8    7    7    8    7    7    8    8    8     8     5     3     6     7     8
S4     8    6    6    7    6    5    6    5    5     5     4     2     5     6     3
S5     8    4    4    6    6    4    3    5    6     4     5     8     5     6     5
S6     7    8    5    6    5    4    3    4    6     3     3     3     5     6     7
S7     8    7    7    7    7    6    7    8    7     6     6     6     8     7     5
S8     3    5    6    5    6    4    6    3    7     7     3     4     4     7     8
S9     7    5    7    6    3    5    5    3    5     4     6     4     5     5     6
S10    3    5    4    1    3    2    3    3    1     2     3     4     4     8     6
S11    7    7    8    6    7    6    6    5    7     5     7     7     7     8     8
S12    6    3    6    3    5    5    3    6    3     0     3     0     1     2     3
S13    4    3    2    4    4    1    4    4    7     4     2     5     4     5     8
S14    7    3    4    4    4    6    7    4    3     2     3     6     8     8     7
```

Recall varies by serial position



Multiple theories of what produces serial position

1. One explanation was primacy (for the first few words) and recency (for the last few words)
2. The middle words were said to be held in some short term memory buffer.
3. Subsequent research showed that memory for events also show serial position effects
4. Football games over a season - tend to remember the first few and the last few.
5. Therefore the lower level of the serial position can not be measuring just short term memory.

Why do we examine the serial position effects?

1. We are seeing if we can produce false memories. But if we do, maybe we are not doing a standard memory task.
2. That is, if we show that people have false memories, some people might say that this is a weird task.
3. But the recall was just as we would have expected.

Preliminaries

1. In an Excel spreadsheet, I had randomly ordered the recognition list.
2. But I kept that information to use later.
3. In R I saved the information in a form that we can read it in easily.
4. This was using the `dput` command
5. To read it back in, just make something equal to what we dput
6. The recognition words were 5 types:
   6.1 Words that were cued by the lists but not presented
   6.2 The first word from each presented list
   6.3 The 8th word from each list
   6.4 The 10th word from each list
   6.5 Control words

The word position information

```
pos.info <- structure(list(word = c("anger", "black", "bread", "chair", "cold",
"doctor", "foot", "fruit", "mad", "white", "butter", "table",
"hot", "nurse", "shoe", "apple", "fight", "death", "dough", "cushion",
"weather", "patient", "arm", "cherry", "wrath", "grief", "flour",
"sofa", "chilly", "physician", "walk", "banana", "girl", "high",
"king", "man", "mountain", "music", "needle", "river", "boy",
"low", "queen", "woman", "hill", "note", "thread", "water", "beautiful",
"cliff", "rule", "beard", "goat", "instrument", "thorn", "barge",
"niece", "building", "throne", "strong", "plain", "horn", "thimble",
"flow"), recog.pos = c(33L, 51L, 48L, 27L, 28L, 13L, 26L, 2L,
17L, 9L, 24L, 63L, 41L, 38L, 43L, 58L, 64L, 25L, 15L, 16L, 21L,
50L, 18L, 4L, 52L, 23L, 49L, 14L, 31L, 34L, 54L, 6L, 44L, 8L,
60L, 32L, 45L, 20L, 35L, 59L, 53L, 29L, 47L, 56L, 55L, 39L, 37L,
10L, 3L, 62L, 22L, 36L, 19L, 57L, 42L, 40L, 61L, 46L, 12L, 5L,
1L, 7L, 30L, 11L), cue = c(1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L,
1L, 1L, 1L, 1L, 1L, 1L, 1L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 0L, 0L, 0L, 0L, 0L), High = c(0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L,
2L, 2L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 0L, 0L, 0L, 0L, 0L, 0L,
0L, 0L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 1L, 2L, 2L, 2L, 2L, 2L, 2L,
2L, 2L, 3L, 3L, 3L, 3L, 3L, 3L, 3L, 3L)), class = "data.frame",
row.names = c(1L,
2L, 3L, 4L, 5L, 6L, 7L, 8L, 9L, 12L, 15L, 18L, 21L, 24L, 27L,
30L, 10L, 13L, 16L, 19L, 22L, 25L, 28L, 31L, 11L, 14L, 17L, 20L,
23L, 26L, 29L, 32L, 33L, 34L, 35L, 36L, 37L, 38L, 39L, 40L, 41L,
44L, 47L, 50L, 53L, 56L, 59L, 62L, 42L, 45L, 48L, 51L, 54L, 57L,
60L, 63L, 43L, 46L, 49L, 52L, 55L, 58L, 61L, 64L))
```

The pos.info object is used to point to the correct items

R code

```
pos.info[c(1:4,9:12,17:20,25:28,60:64),]
```

```
      word recog.pos cue High
1    anger        33   1    0
2    black        51   1    0
3    bread        48   1    0
4    chair        27   1    0
9      mad        17   1    1
12   white         9   1    1
15  butter        24   1    1
18   table        63   1    1
10   fight        64   1    2
13   death        25   1    2
16   dough        15   1    2
19 cushion        16   1    2
11   wrath        52   1    3

14   grief        23   1    3
17   flour        49   1    3
20    sofa        14   1    3
52  strong         5   0    3
55   plain         1   0    3
58    horn         7   0    3
61 thimble        30   0    3
64    flow        11   0    3
```

The recognition data

1. The recognition words were 5 types:
   1.1 Words that were cued by the lists but not presented
   1.2 The first word from each presented list
   1.3 The 8th word from each list
   1.4 The 10th word from list
   1.5 Control words
2. If there were no false memories, we would expect words that were not present to not be recognized
3. We would expect greater recognition for the first than the 8th and 10th words.

More R magic to score the data

We make use of the rowSums function and address particular parts of the data using indirect addressing.

R code

```
Cued <- rowSums(recog[,pos.info$recog.pos[1:8]+1,])/8
First <- rowSums(recog[,pos.info$recog.pos[9:16]+1,])/8
Tenth <- rowSums(recog[,pos.info$recog.pos[17:24]+1,])/8
Eigth <- rowSums(recog[,pos.info$recog.pos[25:32]+1,])/8
Control <- rowSums(recog[,pos.info$recog.pos[33:64]+1,])/32

#put them together into a data.frame
recog.df <- data.frame(First,Eigth,Tenth , Cued,Control)
```

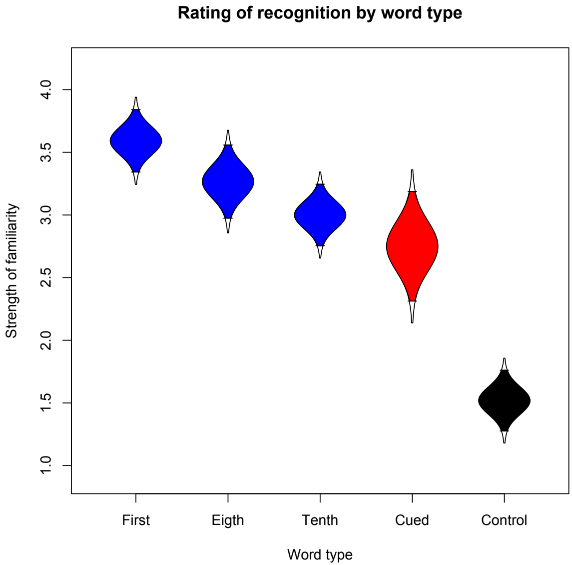
Basic recognition statistics

R code

```
describe(data.df)  #tabular form
#graphic form
error.bars(recog.df,col=c("blue","blue","blue","red","black"),ylab=
"Strength of familiarity",xlab="Word type",
main="Rating of recognition by word type")
```

```
        vars  n mean   sd median trimmed  mad  min  max range  skew kurtosis   se
First      1 15 3.59 0.45   3.88    3.63 0.19 2.62 4.00  1.38 -0.81    -0.90 0.12
Eigth      2 15 3.27 0.53   3.50    3.30 0.37 2.25 3.88  1.62 -0.76    -0.79 0.14
Tenth      3 15 3.00 0.44   3.12    3.06 0.37 1.75 3.50  1.75 -1.33     1.62 0.11
Cued       4 15 2.75 0.79   2.88    2.78 0.93 1.25 3.88  2.62 -0.42    -1.15 0.20
Control    5 15 1.52 0.44   1.41    1.49 0.51 1.03 2.38  1.34  0.53    -1.21 0.11
```

Recognition rating by word type



Descriptive versus Inferential statistics

1. Descriptive statistics
   - Describing the means and variances of data sets allows us to see the results
   - Graphic displays are probably more useful than tables
   - Effect sizes (e.g., Cohen's d) Cohen (1988) allow for comparisons in terms of within group standard errors
   - Effect size = d = $\frac{\bar{x_1} - \bar{x_2}}{sd_{pooled}}$
   - Use the `describe` function to get descriptive statistics
2. Inferential statistics test for probability of the differences
   - Are the observed differences unusual (given chance variation)
   - Student's t (Student, 1908) allows us to estimate the probability of an effect size $p(|d| > 0)$
   - $t = \frac{\bar{x_1} - \bar{x_2}}{\sqrt{\sigma_1^2/N_1 + \sigma_2^2/N_2}}$
   - $t = d\sqrt{N_1 + N_2}$
   - Use `t.test` to find t

Compare the Cued recognition to Controls and presented words

R code

```
t.test(Cued,Control, data=recog.df,paired=TRUE) #Is Cued > Control
t.test(Tenth,Cued,data=recog.df,paired=TRUE)  # do Tenth and Cued differ?
t.test(Eigth,Cued,data=recog.df,paired=TRUE)  #Is Cued < Eigth
```

```
data:  Cued and Control
t = 6.8618, df = 14, p-value = 7.785e-06
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.8464009 1.6160991
sample estimates:
mean of the differences
                1.23125
data:  Tenth and Cued
t = 0.9325, df = 14, p-value = 0.3669
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.3250069  0.8250069
sample estimates:
mean of the differences
                   0.25
data:  Eigth and Cued
t = 2.2389, df = 14, p-value = 0.04192
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.02171807 1.01161526
sample estimates:
mean of the differences
```

Results in words

In a partial replication of Roediger & McDermott (1995) we examined the strength of false recognition for words that were high associates of presented word lists. 15 students were given eight lists of 15 words. Each list was composed of high associates of a non-presented cue word. A brief recall task for the words presented showed the expected serial position effect (Figure 9).

Following the last list, a recognition task was given. Ratings of strength of recognition (definitely did not see, probably did not see, probably saw, definitely saw) were given for the first, eighth and tenth words for each list, as well as the non-presented cue words as well as to 32 non-presented words (control) (Figure 18).

Cued words were falsely recognized ($\bar{x} = 2.75, sd = .79$) more than control words ($\bar{x} = 1.52, sd = .44$), d = 3.67, t = 6.86, p = 7.785e-06, and did not differ from the probability of recognition for the tenth word in each list ($\bar{x} = 3.00$ sd = .44, d = -.50, t = 0.9325, p = .37) but were less than the recognition for the words in the eighth position ($\bar{x} = 3.27$, sd = .53, d = -.1.19, t = 2.23, p

An additional demonstration of false memory

1. Following the 64 item recognition task, a subsequent task asked for True/False recognition of the 8 Cued words and 8 Control words that were high associated of other (non-presented) lists.
2. These responses were the last 16 in the data file.
3. The recognition data file were read in from the remote file and saved as the `recog` object.

Describe the data – Just the last 16 columns

These are the 0/1 of did you see this word. None of these words appeared, but the first 8 were primed by related words.

R code

```
my.data <- recog  #create a new object to match the earlier analysis
describe(my.data[66:81] )  #we specify the column numbers that we want
```

```
describe(my.data[66:81] )  #we specify the column numbers that we want
           vars  n mean   sd median trimmed mad min max range  skew kurtosis   se
anger.1       1 15 0.60 0.51      1    0.62   0   0   1     1 -0.37    -1.98 0.13
black.1       2 15 0.33 0.49      0    0.31   0   0   1     1  0.64    -1.69 0.13
bread.1       3 15 0.53 0.52      1    0.54   0   0   1     1 -0.12    -2.11 0.13
chair.1       4 15 0.53 0.52      1    0.54   0   0   1     1 -0.12    -2.11 0.13
cold.1        5 15 0.40 0.51      0    0.38   0   0   1     1  0.37    -1.98 0.13
doctor.1      6 15 0.27 0.46      0    0.23   0   0   1     1  0.95    -1.16 0.12
foot.1        7 15 0.60 0.51      1    0.62   0   0   1     1 -0.37    -1.98 0.13
fruit.1       8 15 0.53 0.52      1    0.54   0   0   1     1 -0.12    -2.11 0.13
girl.1        9 15 0.00 0.00      0    0.00   0   0   0     0   NaN      NaN 0.00
high.1       10 15 0.07 0.26      0    0.00   0   0   1     1  3.13     8.39 0.07
king.1       11 15 0.07 0.26      0    0.00   0   0   1     1  3.13     8.39 0.07
man.1        12 15 0.13 0.35      0    0.08   0   0   1     1  1.95     1.93 0.09
mountain.1   13 15 0.07 0.26      0    0.00   0   0   1     1  3.13     8.39 0.07
music.1      14 15 0.07 0.26      0    0.00   0   0   1     1  3.13     8.39 0.07
needle.1     15 15 0.07 0.26      0    0.00   0   0   1     1  3.13     8.39 0.07
river.1      16 15 0.13 0.35      0    0.08   0   0   1     1  1.95     1.93 0.09
```

Note that R renamed these variables because they had already appeared as column names.

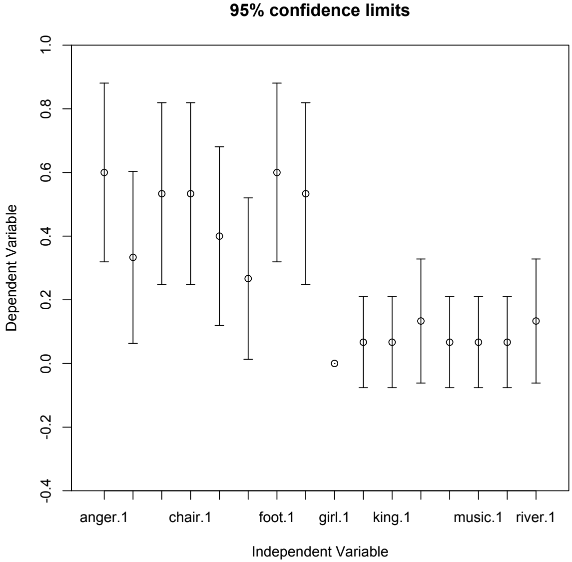
Can also show this graphically

Here we explore 4 different ways of drawing the results.

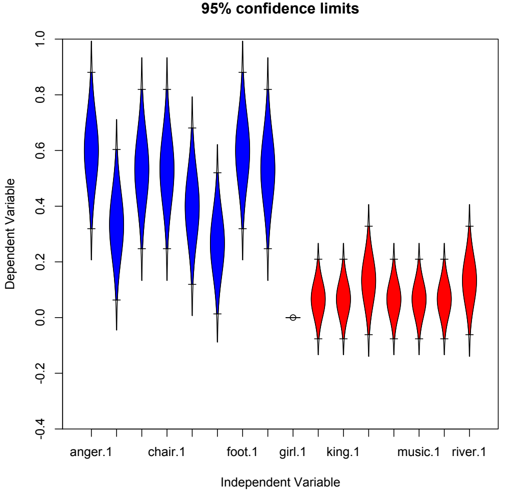
R code

```
error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)),
                               eyes=FALSE) #the basic plot
error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)))
  #with 'cats eyes"  and somewhat improved by specifying x and y axes

error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)),
   ylab=" Recognition",xlab="Prompt", las=2,
     main="False Recognition means and 95% confidence") #means with confidence
 error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)),
   ylab=" Recognition",xlab="Prompt", las=2,
   main="False Recognition means and 95% confidence", eyes=FALSE)

error.dots(my.data[66:81]) #another way to show the results
```

The False Recognition results and their confidence

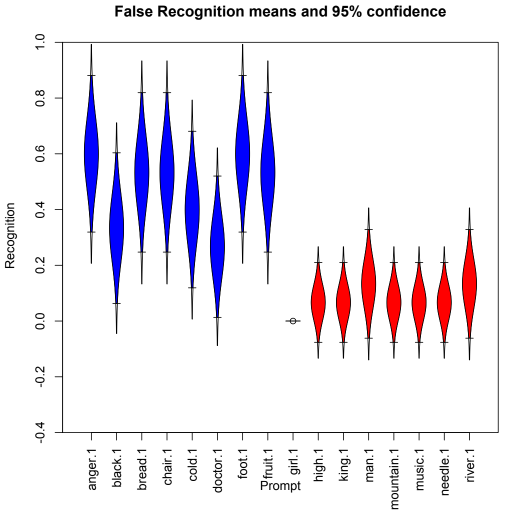


The False Recognition results and their confidence with "cats eyes" to show confidence



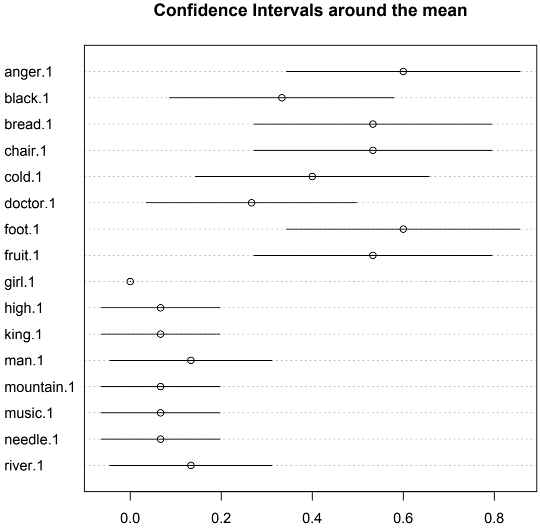

```
error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)))  #the basic plot with colored cats eyes
```

The False Recognition results and their confidence with "cats eyes" to show confidence – improved




```
error.bars(my.data[66:81],col=c(rep("blue",8),rep("red",8)),ylab=" Recognition",xlab="Prompt",
 las=2, main="False Recognition means and 95% confidence")
```

The False Recognition results and their confidence using error.dots



Create some summary data, adjust for number of trials

We can do basic arithmetic on our data to combine the columns in a useful manner.

R code

```
Cued <- rowSums(my.data[66:73])/8 #how many false recognitions that were cued
Control <- rowSums(my.data[74:81]) /8 #How many false recognitions for non-cued
CuedA <- rowSums(my.data[c(66,69,71,72)])/4 #the eyes open condition for group
CuedB <- rowSums(my.data[c(67,68,70,72)])/4 #the eyes open condition for group
Diff = Control - Cued
#Save these as a  data.frame
data.df <- data.frame(group = my.data[,"group"],Cued=Cued,Control=Control,
       CuedA=CuedA,CuedB=CuedB, Diff=Diff)
data.df  #show the data
error.bars.by(CuedA + CuedB + Control ~ group, data=data.df) #draw iit
```

Show the data.frame

R code

```
data.df
```

```
 data.df
   group  Cued Control CuedA CuedB   Diff
1      1 0.250   0.000  0.50  0.25 -0.250
2      2 1.000   0.125  1.00  1.00 -0.875
3      1 0.875   0.125  1.00  1.00 -0.750
4      2 0.000   0.000  0.00  0.00  0.000
5      2 0.875   0.000  0.75  1.00 -0.875
6      1 0.250   0.000  0.25  0.25 -0.250
7      2 0.250   0.000  0.50  0.25 -0.250
8      2 0.625   0.000  0.75  0.50 -0.625
9      2 0.000   0.000  0.00  0.00  0.000
10     1 0.000   0.000  0.00  0.00  0.000
11     1 0.750   0.625  0.50  0.75 -0.125
12     1 0.625   0.000  0.50  0.50 -0.625
13     2 1.000   0.000  1.00  1.00 -1.000
14     2 0.000   0.000  0.00  0.00  0.000
15     2 0.625   0.250  0.75  0.50 -0.375
```

Descriptive statistics of the results

R code

```
describe(data.df)
```

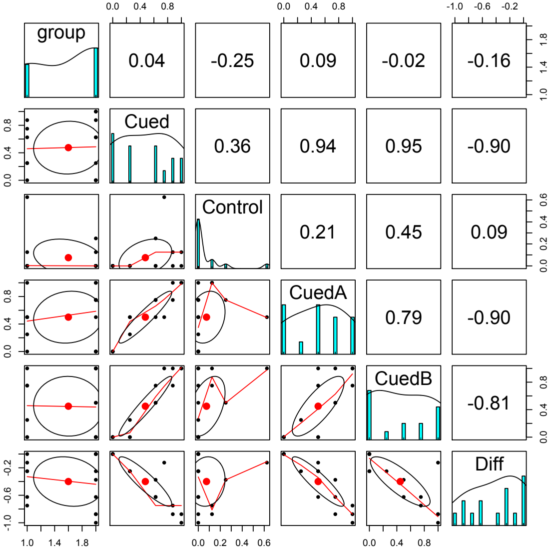
	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
group	1	15	1.60	0.51	2.00	1.62	0.00	1	2.00	1.00	-0.37	-1.98	0.13
Cued	2	15	0.48	0.38	0.62	0.47	0.56	0	1.00	1.00	-0.02	-1.71	0.10
Control	3	15	0.07	0.17	0.00	0.04	0.00	0	0.62	0.62	2.31	4.58	0.04
CuedA	4	15	0.50	0.38	0.50	0.50	0.37	0	1.00	1.00	-0.12	-1.52	0.10
CuedB	5	15	0.47	0.40	0.50	0.46	0.74	0	1.00	1.00	0.20	-1.62	0.10

But it is also useful to show it graphically. We use the `pairs.panels` function to do this.

R code

```
pairs.panels(data.df)
```

A graphical display of our data using `pairs.panels`



Recognition by condition for cued and uncued words

How likely are these differences between Cued and Control due to chance? The use of Student's t test

1. We can test how likely these differences between Cued words and uncued (control words) are by comparing their means to the standard deviations.
2. Mean Cued = .48 with standard deviation of .38
3. Mean Control = .07 with standard deviation of .17
4. Student's "t" = $\frac{\bar{x_1} - \bar{x_2}}{\sqrt{\sigma_1^2/N_1 + \sigma_2^2/N_2}}$ = 3.69
5. Pooled standard deviation $= \sqrt{\frac{N_1 \sigma_1^2 + N_2 * \sigma_2^2}{N_1 + N_2}}$
6. Effect size = d = $\frac{\bar{x_1} - \bar{x_2}}{sd_{pooled}}$ is not sensitive to sample size
7. But t is sensitive to sample size: $t = d\sqrt{N_1 + N_2}$
8. The recommendation is to always report Effect size (e.g., Cohen's d) as well as the t test value Cohen (1988)

Using R to find the t value: two ways

R code

```
t.test(Cued, Control, data= data.df,var.equal=TRUE)
```

```
     Two Sample t-test

data:  Cued and Control
t = 3.6892, df = 28, p-value = 0.0009605
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.1779017 0.6220983
sample estimates:
mean of x mean of y
    0.475     0.075
```

R code

```
t2d(3.689,28)   #convert t to d
```

```
1.39
```

R code

```
t.test(Cued, Control, data= data.df,paired=TRUE)
```

```
        Paired t-test

data:  Cued and Control
t = 4.2982, df = 14, p-value = 0.0007362
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.2003998 0.5996002
sample estimates:
mean of the differences
                    0.4
t2d(4.297,14) = 2.29 (Cohen's d)
```

Interpreting the t-test in English

1. Formally: The probability of their being no difference between the Cued and the Control condition was very small (p = .0009 or p = .0007)
2. Or, more normally: Recognition for cued words (mean = .475, sd = .38) was much greater than the recognition for uncued words (mean =.07, sd = .17) with paired $t_{14} = 4.30$, $p < 0.0007362$ with an effect size (Cohen, 1988) of 2.29.
3. Words that were not presented but were high frequency associates of word lists that were presented were much likely to be falsely reocognized (mean = .475, sd = .38) than were control words that were high frequency associates of word lists that were not presented (mean =.07, sd = .17), paired $t_{14} = 4.30$, $p < 0.0007362$ with an effect size (Cohen, 1988) of 2.29.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed ed.). Hillsdale, N.J.: L. Erlbaum Associates.

Roediger, H. L. & McDermott, K. B. (1995). Creating false memories: Remembering words not presented in lists. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *21*(4), 803–814.

Student (1908). The probable error of a mean. *Biometrika*, *6*(1), 1–25.