BSPv2

Aleksander Nitka

25/09/2018

List matlab files in directory

```
# Change that to wherever your data is
matdir = "/Users/aleksandernitka/OneDrive\ -\ The\ University\ of\ Nottingham/Lehigh_2018/Data/matlab_da

files.mat = list.files(path = matdir, pattern = "\\.mat$")
print(files.mat)
```

```
##  [1] "BSPv2_1_block1_cond2.mat"  "BSPv2_1_block2_cond0.mat"
##  [3] "BSPv2_1_block3_cond1.mat"  "BSPv2_12_block1_cond1.mat"
##  [5] "BSPv2_12_block2_cond0.mat" "BSPv2_12_block3_cond2.mat"
##  [7] "BSPv2_13_block1_cond2.mat" "BSPv2_13_block2_cond1.mat"
##  [9] "BSPv2_13_block3_cond0.mat" "BSPv2_14_block1_cond1.mat"
## [11] "BSPv2_14_block2_cond0.mat" "BSPv2_14_block3_cond2.mat"
## [13] "BSPv2_15_block1_cond2.mat" "BSPv2_15_block2_cond1.mat"
## [15] "BSPv2_15_block3_cond0.mat" "BSPv2_16_block1_cond0.mat"
## [17] "BSPv2_16_block2_cond1.mat" "BSPv2_16_block3_cond2.mat"
## [19] "BSPv2_17_block1_cond1.mat" "BSPv2_17_block2_cond0.mat"
## [21] "BSPv2_17_block3_cond2.mat" "BSPv2_18_block1_cond1.mat"
## [23] "BSPv2_18_block2_cond2.mat" "BSPv2_18_block3_cond0.mat"
## [25] "BSPv2_19_block1_cond0.mat" "BSPv2_19_block2_cond1.mat"
## [27] "BSPv2_19_block3_cond2.mat" "BSPv2_2_block1_cond0.mat"
## [29] "BSPv2_2_block2_cond2.mat"  "BSPv2_2_block3_cond1.mat"
## [31] "BSPv2_20_block1_cond1.mat" "BSPv2_20_block2_cond0.mat"
## [33] "BSPv2_20_block3_cond2.mat" "BSPv2_3_block1_cond0.mat"
## [35] "BSPv2_3_block2_cond2.mat"  "BSPv2_3_block3_cond1.mat"
## [37] "BSPv2_4_block1_cond0.mat"  "BSPv2_4_block2_cond1.mat"
## [39] "BSPv2_4_block3_cond2.mat"  "BSPv2_5_block1_cond1.mat"
## [41] "BSPv2_5_block2_cond2.mat"  "BSPv2_5_block3_cond0.mat"
## [43] "BSPv2_6_block1_cond2.mat"  "BSPv2_6_block2_cond1.mat"
## [45] "BSPv2_6_block3_cond0.mat"
```

Load files into a one data frame

```
for (i in 1:length(files.mat)) {

    # Extract details from file name
    exp = strsplit(files.mat[i], "_")[[1]][1]
    ss = strsplit(files.mat[i], "_")[[1]][2]
    block = strsplit(files.mat[i], "_")[[1]][3]
    cond = strsplit(strsplit(files.mat[i], "_")[[1]][4], '[.]')[[1]][1]

    if (nchar(ss) == 1){
        ss = paste("0", ss, sep = '')
    }
```

```
    # Read mat file
    dat = readMat( paste(matdir, files.mat[i], sep = "/"))

    # Extract data from mat to R
    # searchRT - rection time to non-porbe trials, 0 if probe trial but 0 also if no response
    # targetletters - no of accurately reported letters on target side
    # distletters - no of accurately reported letters on non-target side
    # probe - logical if trial was a probe or not
    # SearchRespCode - 38 is UpArrow and 40 is DownArrow



    tmp = as.data.frame(cbind(t(dat$searchRT),
                              t(dat$searchRespCode),
                              t(dat$searchacc),
                              t(dat$targletters),
                              t(dat$distletters),
                              dat$probe,
                              t(dat$FA),
                              t(dat$colorcue),
                              exp, ss, block, cond
                              ))

    names(tmp) = c('RT', 'SearchRespCode', 'SearchAccuracy', 'TargLetters', 'DistLetters', 'ProbeTrial'
                   'ColorCue', 'Exp', 'SS', 'Block', 'Cond')

    # Extract orientation codes
    orient = as.data.frame(dat$Orient)
    names(orient) = c('orient01', 'orient02', 'orient03', 'orient04',
                      'orient05', 'orient06', 'orient07', 'orient08',
                      'orient09', 'orient10', 'orient11', 'orient12')

    #tmp = cbind(tmp, orient)

    if (i == 1){
        # Create main data frame by copying the tmp
        BSPv2 = tmp
        print('BSPv2 data frame created')
        print(sprintf('Added ss %s, %s, %s to BSPv2', ss, block, cond))
        remove(tmp, block, cond, exp, ss, orient, dat)
    }
    else {
        # bind main data frame and tmp
        BSPv2 = rbind(BSPv2, tmp)
        print(sprintf('Added ss %s, %s, %s to BSPv2', ss, block, cond))
        remove(tmp, block, cond, exp, ss, orient, dat)

    }

}
```

```
## [1] "BSPv2 data frame created"
## [1] "Added ss 01, block1, cond2 to BSPv2"
```

```
## [1] "Added ss 01, block2, cond0 to BSPv2"
## [1] "Added ss 01, block3, cond1 to BSPv2"
## [1] "Added ss 12, block1, cond1 to BSPv2"
## [1] "Added ss 12, block2, cond0 to BSPv2"
## [1] "Added ss 12, block3, cond2 to BSPv2"
## [1] "Added ss 13, block1, cond2 to BSPv2"
## [1] "Added ss 13, block2, cond1 to BSPv2"
## [1] "Added ss 13, block3, cond0 to BSPv2"
## [1] "Added ss 14, block1, cond1 to BSPv2"
## [1] "Added ss 14, block2, cond0 to BSPv2"
## [1] "Added ss 14, block3, cond2 to BSPv2"
## [1] "Added ss 15, block1, cond2 to BSPv2"
## [1] "Added ss 15, block2, cond1 to BSPv2"
## [1] "Added ss 15, block3, cond0 to BSPv2"
## [1] "Added ss 16, block1, cond0 to BSPv2"
## [1] "Added ss 16, block2, cond1 to BSPv2"
## [1] "Added ss 16, block3, cond2 to BSPv2"
## [1] "Added ss 17, block1, cond1 to BSPv2"
## [1] "Added ss 17, block2, cond0 to BSPv2"
## [1] "Added ss 17, block3, cond2 to BSPv2"
## [1] "Added ss 18, block1, cond1 to BSPv2"
## [1] "Added ss 18, block2, cond2 to BSPv2"
## [1] "Added ss 18, block3, cond0 to BSPv2"
## [1] "Added ss 19, block1, cond0 to BSPv2"
## [1] "Added ss 19, block2, cond1 to BSPv2"
## [1] "Added ss 19, block3, cond2 to BSPv2"
## [1] "Added ss 02, block1, cond0 to BSPv2"
## [1] "Added ss 02, block2, cond2 to BSPv2"
## [1] "Added ss 02, block3, cond1 to BSPv2"
## [1] "Added ss 20, block1, cond1 to BSPv2"
## [1] "Added ss 20, block2, cond0 to BSPv2"
## [1] "Added ss 20, block3, cond2 to BSPv2"
## [1] "Added ss 03, block1, cond0 to BSPv2"
## [1] "Added ss 03, block2, cond2 to BSPv2"
## [1] "Added ss 03, block3, cond1 to BSPv2"
## [1] "Added ss 04, block1, cond0 to BSPv2"
## [1] "Added ss 04, block2, cond1 to BSPv2"
## [1] "Added ss 04, block3, cond2 to BSPv2"
## [1] "Added ss 05, block1, cond1 to BSPv2"
## [1] "Added ss 05, block2, cond2 to BSPv2"
## [1] "Added ss 05, block3, cond0 to BSPv2"
## [1] "Added ss 06, block1, cond2 to BSPv2"
## [1] "Added ss 06, block2, cond1 to BSPv2"
## [1] "Added ss 06, block3, cond0 to BSPv2"
```

Do some data manipulations, prep for mean analysis

```
# data type must be changed from integer to numeric, but converting directly changes the values,
# must convert to string and then to numeric to keep the actual values
BSPv2$RT = as.numeric(as.character(BSPv2$RT))
BSPv2$TargLetters = as.numeric(BSPv2$TargLetters)
BSPv2$DistLetters = as.numeric(BSPv2$DistLetters)
BSPv2$TrialType = 'RT'
```

```
BSPv2$TrialType[BSPv2$ProbeTrial == 1] = 'PR'
BSPv2$SS = as.numeric(as.character(unique(BSPv2$SS)))

# If RT = 0 and trial != Probe then it was a timeOut
# need to mark those as otherwise 0 will be counted towards the mean
BSPv2$timeOut = 0
BSPv2$timeOut[BSPv2$RT == 0 & BSPv2$ProbeTrial == 0] = 1

# Get subject IDs
SSid = unique(BSPv2$SS)

# Create a DF for means
means_BSPv2 = data.frame(matrix(nrow = length(SSid)*3, ncol = 5))
names(means_BSPv2) = c('SS', 'Cond', 'mRT', 'LettersTarget', 'LettersDistractor')
means_BSPv2$SS = SSid
means_BSPv2$Cond = c(rep('cond0',length(SSid)), rep('cond1',length(SSid)), rep('cond2',length(SSid)))
means_BSPv2$Accuracy = NA # 1 means correct. 0 mean incorrect and 2 means no response or participant ha
```

Extract means

```
for (i in 1:nrow(means_BSPv2)){

    # isolate 1 ss data
    d = subset(BSPv2, BSPv2$SS == means_BSPv2$SS[i])
    # isolate rt trials
    r = subset(d, d$TrialType == 'RT')
    # isolate probe trials
    p = subset(d, d$TrialType == 'PR')

    # calculate mean RT form a subset of condition, exclude Timed out trials and Incorrect responses
    means_BSPv2$mRT[i] = mean( subset( r$RT, r$Cond == means_BSPv2$Cond[i] & r$timeOut == 0 & r$SearchA

    # calculate mean Accuracy for condition, form: (timeout + inaccurate) / all trials
    means_BSPv2$Accuracy[i] = 1 - (nrow(subset(r, r$SearchAccuracy != 1 & r$Cond == means_BSPv2$Cond[i]

    # calcualte the mean number of letters reported on Target side in each probe trial
    means_BSPv2$LettersTarget[i] = mean(subset( p$TargLetters, p$Cond == means_BSPv2$Cond[i] ))

    # do the same for distracotr letters
    means_BSPv2$LettersDistractor[i] = mean(subset( p$DistLetters, p$Cond == means_BSPv2$Cond[i] ))



}

# Make sure that data types as good
means_BSPv2$Cond = as.factor(means_BSPv2$Cond)
means_BSPv2$LettersTarget = as.numeric(means_BSPv2$LettersTarget)
means_BSPv2$LettersDistractor = as.numeric(means_BSPv2$LettersDistractor)


# Probe trials, Create Difference Index = Target - Distractor
```

```
means_BSPv2$LettersDiff = means_BSPv2$LettersTarget - means_BSPv2$LettersDistractor
```

Individual Means

SS	Cond	mRT	LettersTarget	LettersDistractor	Accuracy	LettersDiff
1	cond0	1.737111	2.238095	2.142857	0.8823529	0.0952381
12	cond0	1.565233	2.666667	2.000000	0.8500000	0.6666667
13	cond0	1.819340	1.750000	2.312500	0.8852459	-0.5625000
14	cond0	1.773017	2.484849	1.909091	0.9193548	0.5757576
15	cond0	1.992718	2.066667	2.266667	0.8153846	-0.2000000
16	cond0	1.788464	2.214286	2.357143	0.9104478	-0.1428571
17	cond0	1.878958	2.393939	2.272727	0.8750000	0.1212121
18	cond0	1.826503	2.483871	2.096774	0.8787879	0.3870968
19	cond0	1.895461	1.880000	2.280000	0.8472222	-0.4000000
2	cond0	1.864198	2.111111	2.305556	0.8095238	-0.1944444
20	cond0	1.797548	2.000000	2.272727	0.8636364	-0.2727273
3	cond0	2.003652	2.185185	2.148148	0.8055556	0.0370370
4	cond0	1.777396	2.354839	1.870968	0.7846154	0.4838710
5	cond0	1.834614	2.142857	2.285714	0.8360656	-0.1428571
6	cond0	1.783754	2.096774	2.225807	0.8923077	-0.1290323
1	cond1	1.556350	2.722222	1.555556	0.8666667	1.1666667
12	cond1	1.527664	2.413793	1.689655	0.9104478	0.7241379
13	cond1	1.548140	2.281250	1.656250	0.9062500	0.6250000
14	cond1	1.763101	2.128205	1.948718	0.8448276	0.1794872
15	cond1	1.721742	2.105263	1.842105	0.9491525	0.2631579
16	cond1	1.778432	2.200000	1.840000	0.9027778	0.3600000
17	cond1	1.696948	2.312500	1.718750	0.9062500	0.5937500
18	cond1	1.716949	2.357143	1.750000	0.8529412	0.6071429
19	cond1	1.709725	2.769231	1.564103	0.9473684	1.2051282
2	cond1	1.566890	2.342105	1.842105	0.8947368	0.5000000
20	cond1	1.666259	2.280000	1.680000	0.8857143	0.6000000
3	cond1	1.621925	2.533333	1.633333	0.9076923	0.9000000
4	cond1	1.676746	2.625000	1.468750	0.9218750	1.1562500
5	cond1	1.547659	2.281250	1.625000	0.9062500	0.6562500
6	cond1	1.657899	2.480000	1.680000	0.8873239	0.8000000
1	cond2	1.293611	2.870968	1.161290	0.9264706	1.7096774
12	cond2	1.242669	3.000000	1.241379	0.9571429	1.7586207
13	cond2	1.416916	3.027778	1.333333	0.9841270	1.6944444
14	cond2	1.281384	2.937500	1.156250	0.9218750	1.7812500
15	cond2	1.429838	3.064516	1.225807	0.9846154	1.8387097
16	cond2	1.415566	3.238095	1.190476	0.9259259	2.0476190
17	cond2	1.340420	2.840000	1.080000	0.9142857	1.7600000
18	cond2	1.327880	3.000000	1.392857	0.9402985	1.6071429
19	cond2	1.384125	3.000000	1.290323	0.9843750	1.7096774
2	cond2	1.390078	3.230769	1.179487	0.9636364	2.0512821
20	cond2	1.394467	3.115385	1.269231	0.9411765	1.8461538
3	cond2	1.323054	2.972222	1.194444	0.9655172	1.7777778
4	cond2	1.266849	2.733333	1.533333	0.9545455	1.2000000
5	cond2	1.369688	2.935484	1.161290	0.8769231	1.7741935
6	cond2	1.322927	3.151515	1.242424	0.9206349	1.9090909

Fast results - descriptives

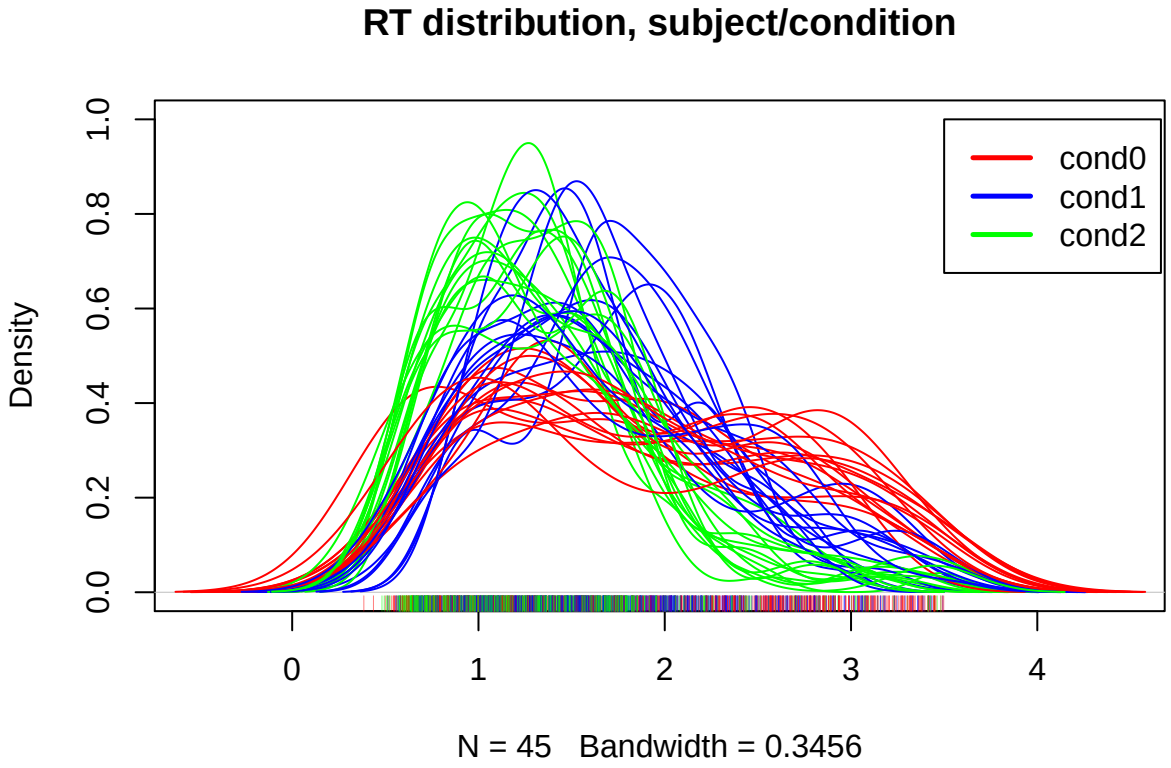
RT analysis

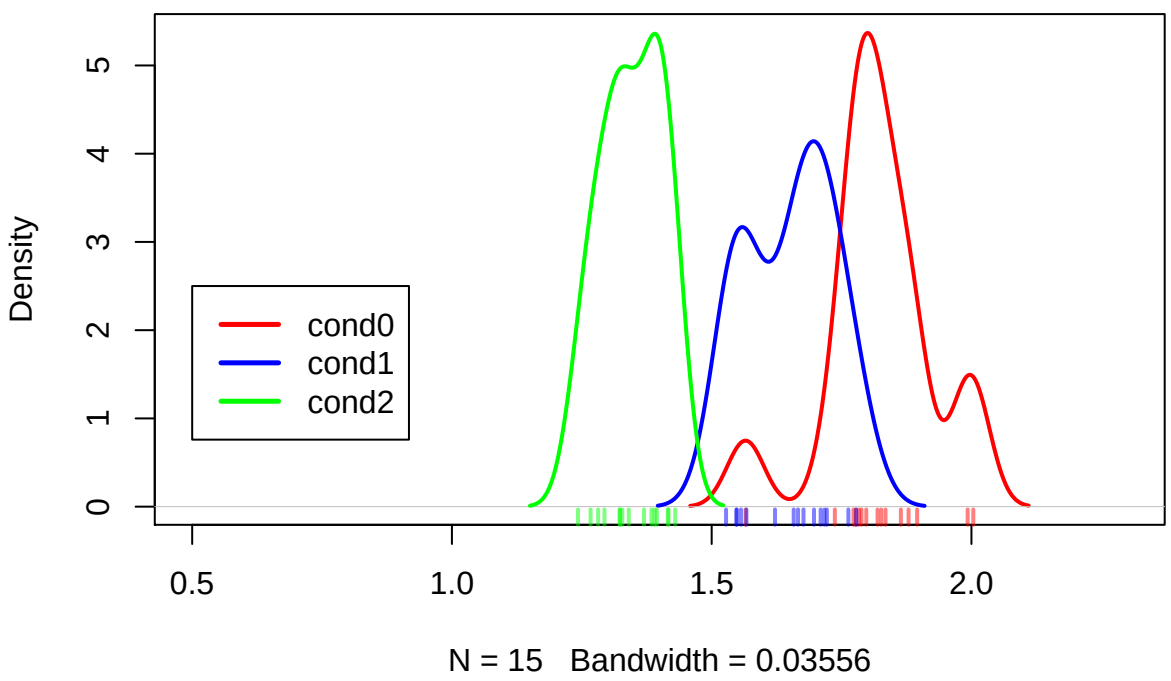
Performed for N = 15. Mean RT for **'cond0'** was 1.8225312 (SD = 0.1046953), for **'cond1'** it was 1.6504285 (SD = 0.083669) and for **'cond2'** it was 1.3466314 (SD = 0.0588449).

Accuracy, the mean accuracy for **'cond0'** was M = 0.8570334 (SD = 0.0402931) and for the **'cond1'** condition M = 0.8993516, SD = (0.0295046) and M = M = 0.9441033, SD = (0.0305843) for **'cond2'**.

Probe analysis.

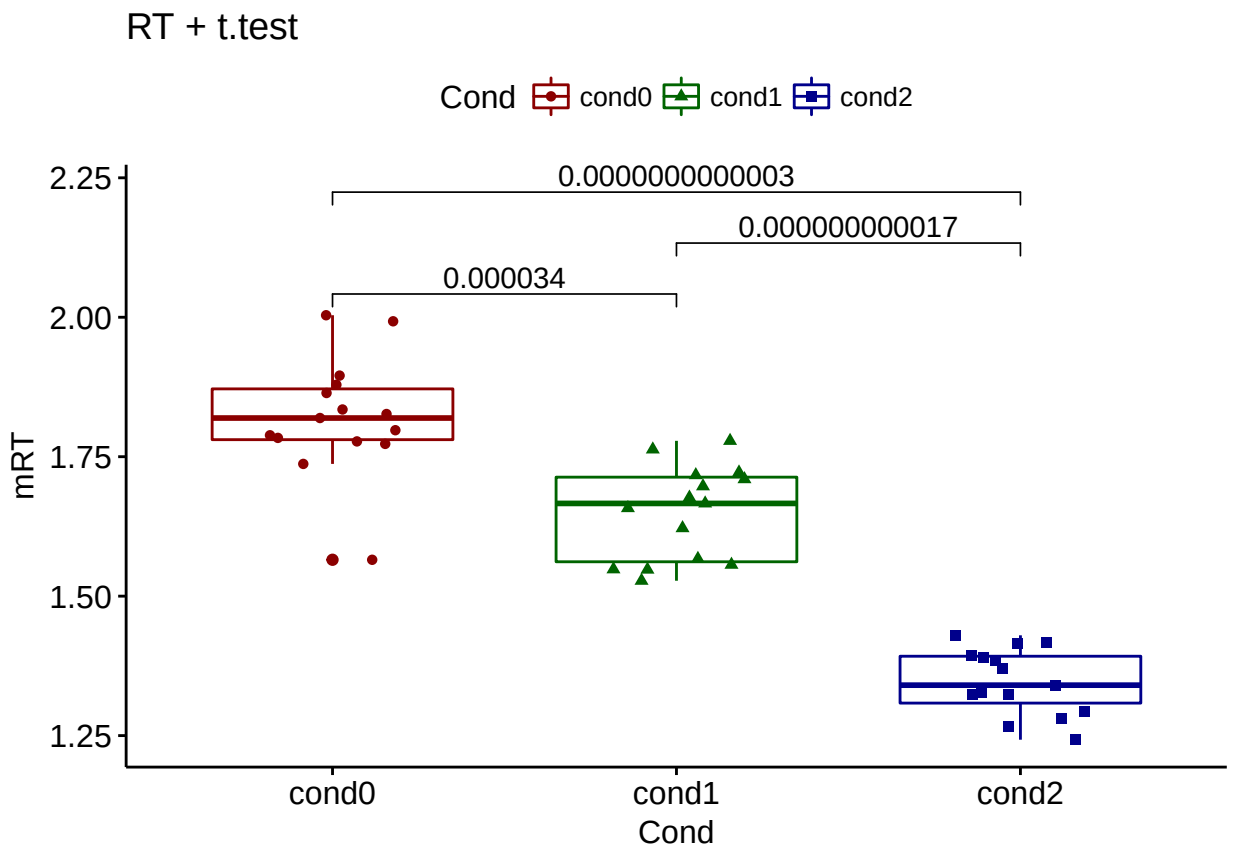
On **'cond0'** trials participants reported M = 2.2046093 (SD = 0.2430671) letters on the Target Side and M = 2.1831119 (SD = 0.151928) on Distractor Side. During the **'cond1'** trials participants reported M = 2.3887531 (SD = 0.2016794) letters on the Target Side and M = 1.6996217 (SD = 0.1282846) on Distractor Side. Finally, for the **'cond2'** trials participants reported M = 3.0078377 (SD = 0.1394959) letters on the Target Side and M = 1.2434617 (SD = 0.1117175) on Distractor Side.



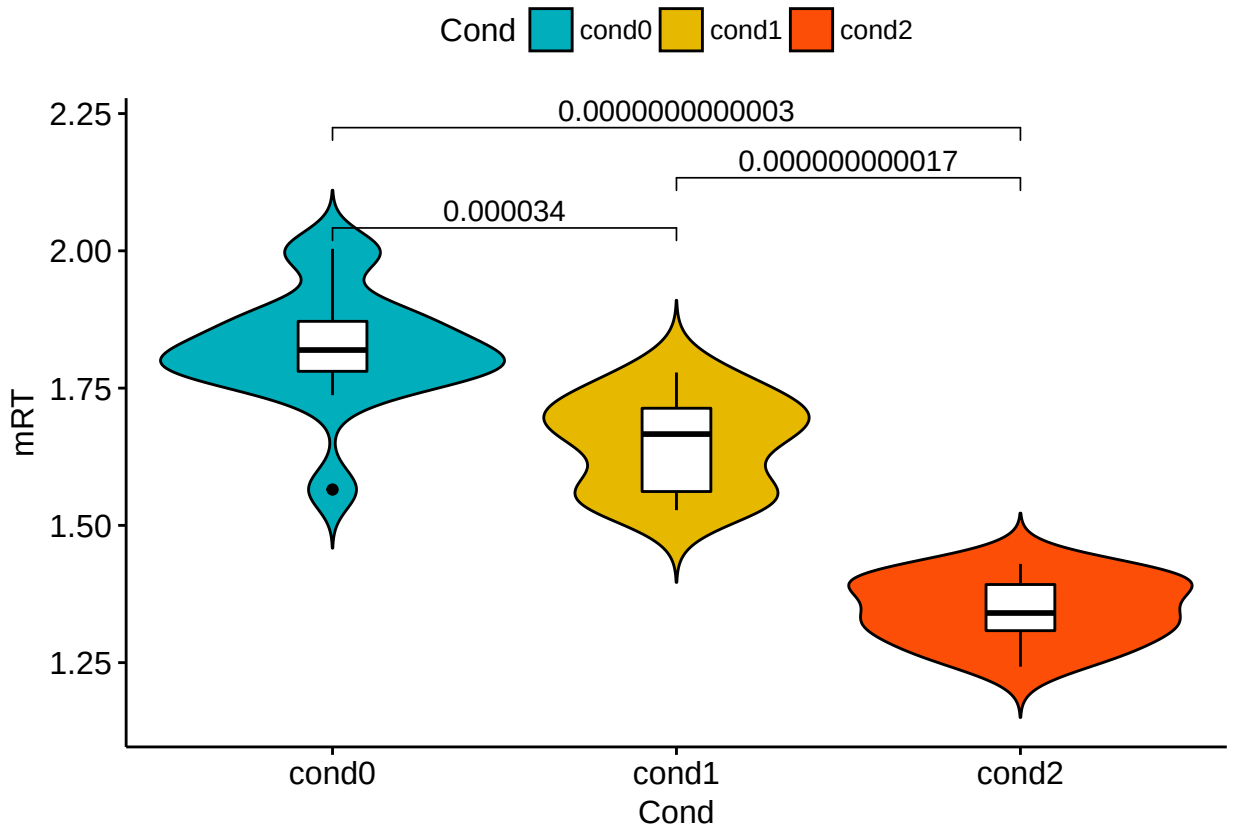



```
#ggplot(data = means_BSPv2, aes(Cond, mRT)) + geom_point(color="skyblue") + stat_summary(fun.y=mean,geo

my_test = list( c("cond0", "cond1"), c("cond1", "cond2"), c("cond0", "cond2") )

rt_plg = ggboxplot(means_BSPv2, x = "Cond", y = "mRT",
```

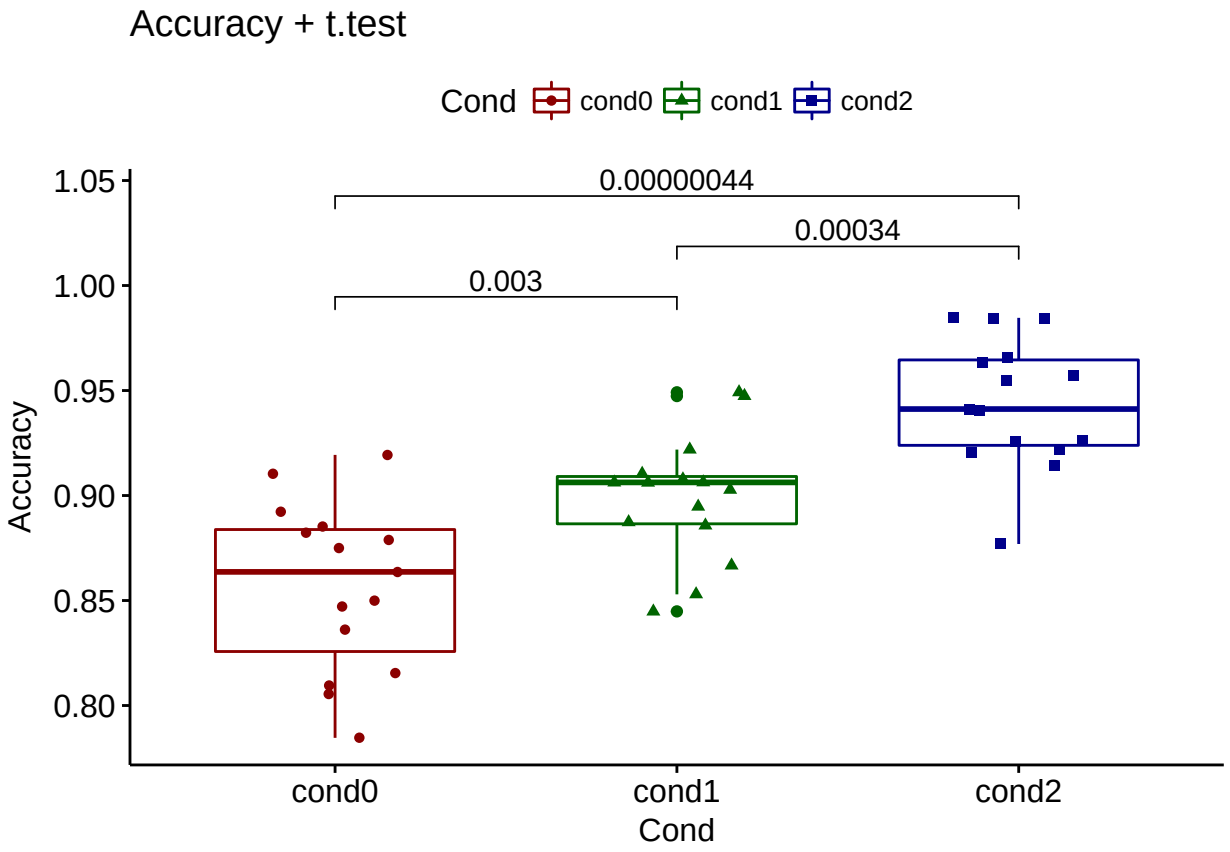
```
                color = "Cond", palette =c("dark red", "dark green", "dark blue"),
                add = "jitter", shape = "Cond") + stat_compare_means(comparisons = my_test, method = "t

rt_plg
```



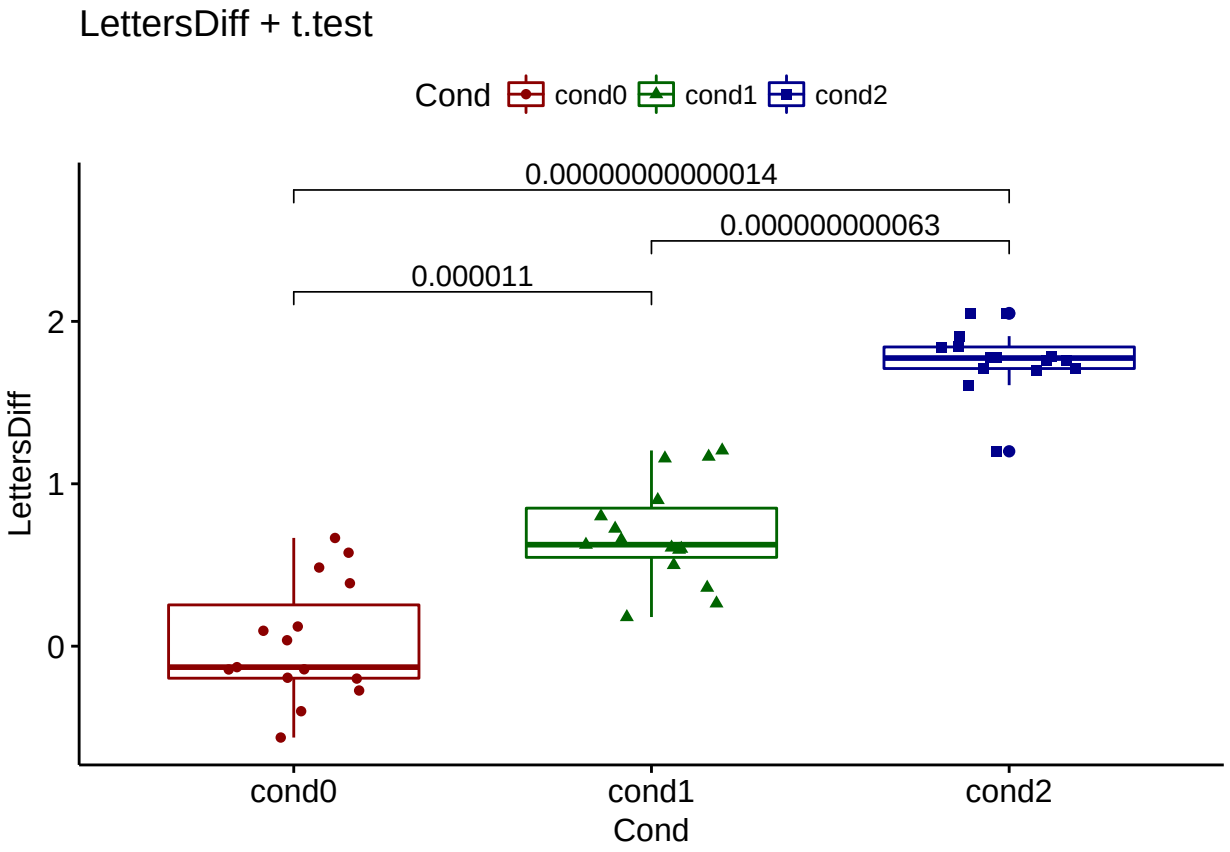

```
rt_violin = ggviolin(means_BSPv2, x = "Cond", y = "mRT", fill = "Cond",
         palette = c("#00AFBB", "#E7B800", "#FC4E07"),
         add = "boxplot", add.params = list(fill = "white"))+ stat_compare_means(comparisons = my_test,

rt_violin
```



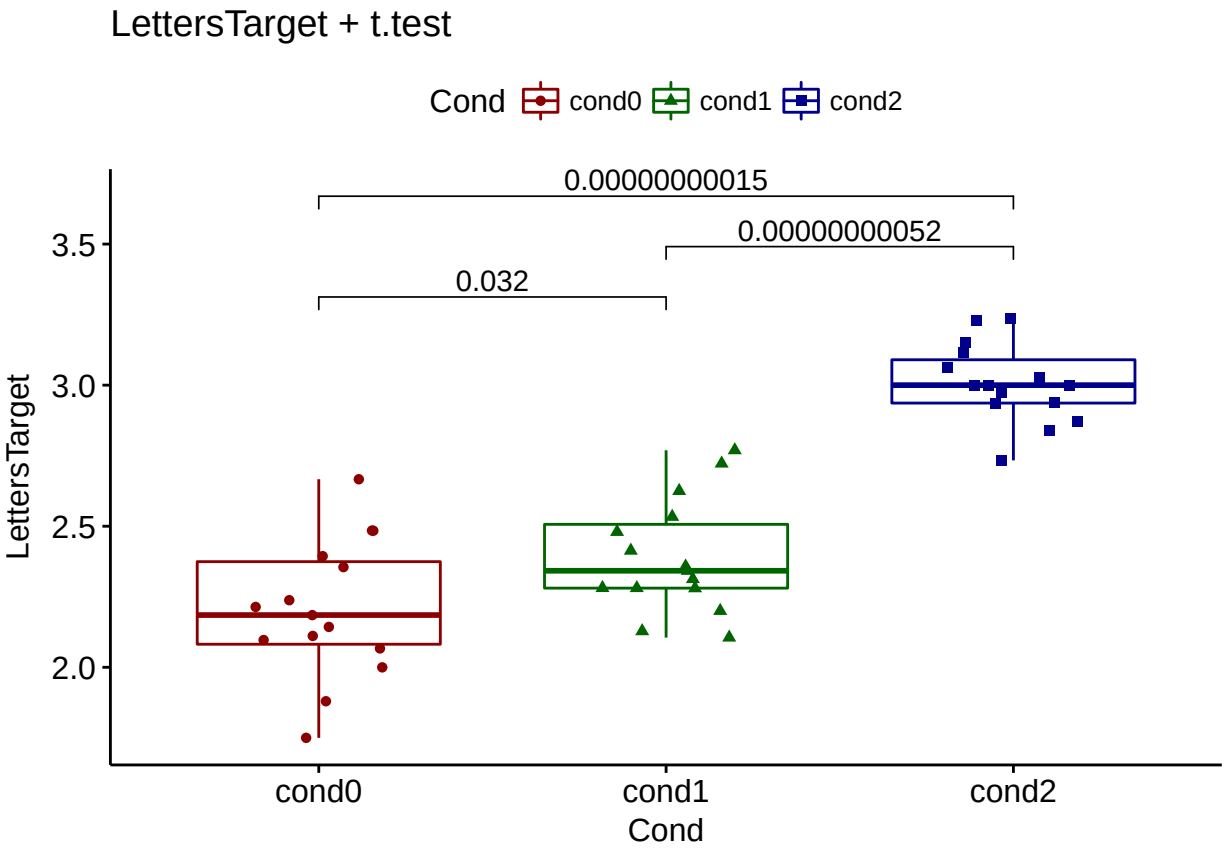

```
my_test = list( c("cond0", "cond1"), c("cond1", "cond2"), c("cond0", "cond2") )

acc_plg = ggboxplot(means_BSPv2, x = "Cond", y = "Accuracy",
                color = "Cond", palette =c("dark red", "dark green", "dark blue"),
                add = "jitter", shape = "Cond") + stat_compare_means(comparisons = my_test, method = "t
acc_plg
```



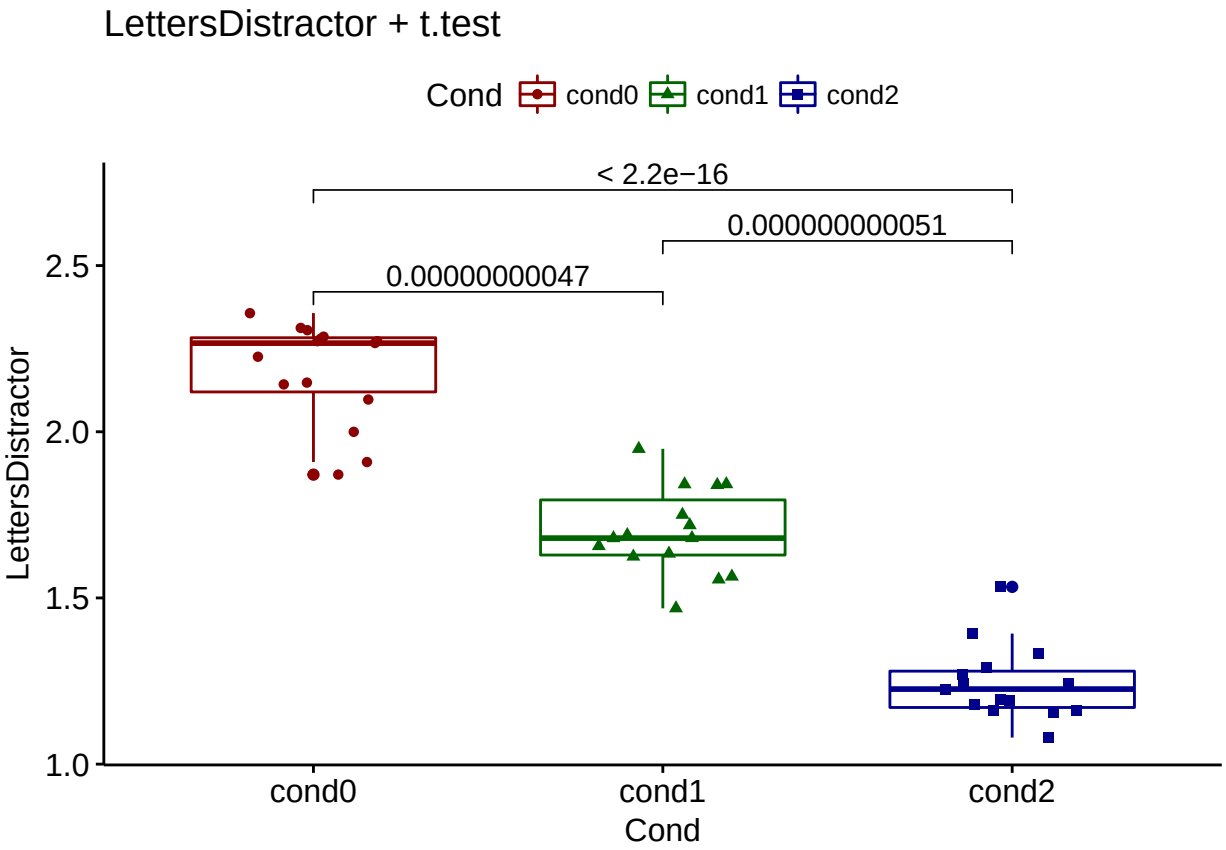

```
my_test = list( c("cond0", "cond1"), c("cond1", "cond2"), c("cond0", "cond2") )

lettDiff_plg = ggboxplot(means_BSPv2, x = "Cond", y = "LettersDiff",
                color = "Cond", palette =c("dark red", "dark green", "dark blue"),
                add = "jitter", shape = "Cond") + stat_compare_means(comparisons = my_test, method = "t
lettDiff_plg
```




```
lettTar_plg = ggboxplot(means_BSPv2, x = "Cond", y = "LettersTarget",
                color = "Cond", palette =c("dark red", "dark green", "dark blue"),
                add = "jitter", shape = "Cond") + stat_compare_means(comparisons = my_test, method = "t
lettTar_plg
```




```
lettDist_plg = ggboxplot(means_BSPv2, x = "Cond", y = "LettersDistractor",
                color = "Cond", palette =c("dark red", "dark green", "dark blue"),
                add = "jitter", shape = "Cond") + stat_compare_means(comparisons = my_test, method = "t
lettDist_plg
```




```
## RT
# normality test - all RT data
rt_sw = shapiro.test(means_BSPv2$mRT)
# as above but for cond0, 1, 2
rt_sw_c0 = shapiro.test(means_BSPv2$mRT[means_BSPv2$Cond == 'cond0'])
rt_sw_c1 = shapiro.test(means_BSPv2$mRT[means_BSPv2$Cond == 'cond1'])
rt_sw_c2 = shapiro.test(means_BSPv2$mRT[means_BSPv2$Cond == 'cond2'])

# Bartlett Test of Homogeneity of Variances PARAMETIC
# From: http://www.instantr.com/2012/12/12/performing-bartletts-test-in-r/
# Bartlett's test allows you to compare the variance of two or more samples to determine whether they a
# NS means we cannot reject the null hypothesis that the variance is the same for all treatment groups.
rt_bt = bartlett.test(mRT~Cond, data=means_BSPv2)

# Levene's test (does the same as Bartlett), if NS homescedascity can be assumed.
rt_lt = leveneTest(mRT~Cond, data=means_BSPv2)
rt_lt_p = rt_lt$`Pr(>F)`[1] # because of stupid packing of this value

# Figner-Killeen Test of Homogeneity of Variances NON-PARAMETIC
# flignerr.test(mRT~Cond, data=means_BSPv2)
```

Assumptions testing, RT

For **normality** of data Shapiro-Wilk test was used. Normality can be assumed if the test is ns and so for all RT data it was p = 0.0309329, for a subset of **'cond0'** RTs it as p = 0.2067814, for **'cond1'** p = 0.2070537 and for the **'cond2'** p = 0.5381294.

Next the assumption that **variances** of the populations from which different samples are drawn are equal. This was done with Bartlett Test of Homogeneity of Variances and Levene's Test. If either of those tests comes out as ns it means we cannot reject the null hypothesis that the variance is the same for all treatment groups. Bartlett test had p = 0.1177068 and Levene's had p = 0.4669794.

```
rt_aov = aov(data = means_BSPv2, mRT ~ Cond)
summary(rt_aov)
```

```
##             Df Sum Sq Mean Sq F value              Pr(>F)
## Cond         2 1.7420  0.8710     122 <0.0000000000000002 ***
## Residuals   42 0.2999  0.0071
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Run this for variances not equal:
#rt_aov_ow = oneway.test(data = means_BSPv2, mRT ~ Cond)
```