Opi_syn_circuit_plots_R

April 29, 2019

```
In [41]: library(IRdisplay)

          display_html(
          '<script>
          code_show=true;
          function code_toggle() {
            if (code_show){
              $(\'div.input\').hide();
            } else {
              $(\'div.input\').show();
            }
            code_show = !code_show
          }
          $( document ).ready(code_toggle);
          </script>
            <form action="javascript:code_toggle()">
              <input type="submit" value="Click here to toggle on/off the raw code.">
           </form>'
          )
```

1 Introduction

Based on data in folder 'data', analysis - plots - stats are ran for the opi-syn-circuit project. For in-depth explanation of the data acquisition and preprocessing read: - matlab_instructions.txt - Birdsong, Jongbloets et al. eLife 2019

```
In [1]: #import packages
         library(ggplot2)
         library(reshape2)
         library(cowplot)
         require(cowplot)
         library(plyr)
         library(Skillings.Mack)
         library(extrafont)
         library(VennDiagram)
         library(Cairo)
         library(dunn.test)
library(scales)

#read required datasets
df=read.csv('data/effectdataset.csv')
df$X=NULL
cdf=read.csv('data/rawdataset.csv') # will be used for latency graphs
# make new column describing the part of the circuit measured.
cdf$X=NULL
cdf$circuit= paste(cdf$stimSource, cdf$recordLayer, cdf$cellType, sep='_')
apdf=read.csv('data/latencySpikePeak.csv')
apdf$X=NULL



#+++++++++++++++++++++++++
# Function to calculate the mean and the standard deviation
  # for each group
#+++++++++++++++++++++++++
# data : a data frame
# varname : the name of a column containing the variable
  #to be summariezed
# groupnames : vector of column names to be used as
  # grouping variables
data_meanSDSEM <- function(data, varname, groupnames){
  require(plyr)
  summary_func <- function(x, col){
    c(n = sum(!is.na(x[[col]])), mean = mean(x[[col]], na.rm=TRUE),
      sd = sd(x[[col]], na.rm=TRUE), sem = sd(x[[col]], na.rm=TRUE)/(sqrt(sum(!is.na(x
  }
  # , sem = sd/(sum(!is.na(x[[col]])))^2
  data_sum<-ddply(data, groupnames, .fun=summary_func,
                  varname)
  data_sum <- rename(data_sum, c("mean" = varname))
 return(data_sum)
}
#++++++++++++++++++++++++++++
#Function to append number of observations
give.n  <- function(x){
    return(c(y = -10, label = length(x)))
# for now it should print the number of observations at -5%;base of the graph
}
#++++++++++++++++++++++++++++
#function to annotate asterisks to plot
pvalAnno <- function(pval,adjust){
                        pval = pval*adjust
            if (!is.finite(pval)){
                pvalS = 'Error'
                fontsize = 11.17
```

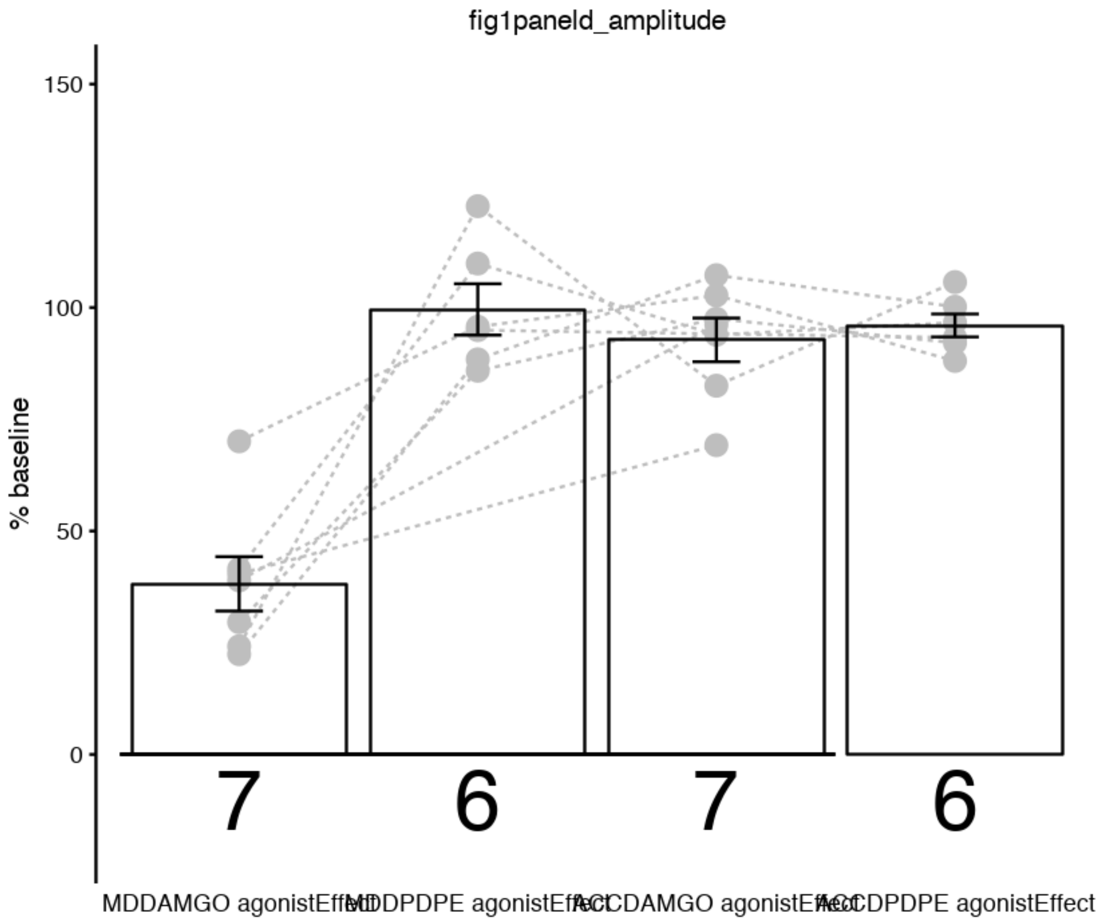
```
                        fontface = 1
                    }
                    else if(pval > 0.05){
                                        pvalS ='N.S.'
                                        fontsize = 11.17
                                        fontface = 1
                                }
                                else if(pval <= 0.05 & pval > 0.01){
                                        pvalS = '*'
                                        fontsize = 13.88
                                        fontface = 2
                                }
                                else if(pval <=0.01 & pval >0.001){
                                        pvalS = '**'
                                        fontsize = 13.88
                                        fontface = 2
                                }
                                else if(pval <=0.001){
                                        pvalS='***'
                                        fontsize = 13.88
                                        fontface = 2
                                }
                return(c(pvalS, fontsize,fontface))
        }
```

```
Attaching package: cowplot

The following object is masked from package:ggplot2:

    ggsave

Registering fonts with R
Loading required package: grid
Loading required package: futile.logger
```

2 Figure 1d

In [2]:
```
########################################################################################
#     Figure 1 panel d
########################################################################################
# Subset: two channel opto of ACC and MD wihtin DMS, recorded EPSC, from MSNs with DAM
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & stimSource == "MD
tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "MDDAMGO" | agonistName ==

#Important for good functioning of the script: No need to change:
```

```
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("MDDAMGO agonistEffect", "MDDPDPE agonistEffect", "ACCDAMGO agonistEffect",'
antagonistSelect = c("ENKWASH")
graphTitle = c("fig1paneld")

k = 1
variableSelect = 'amplitude'
#since we adding more groups to the plot we need to make a plottingColumn to tell what
sDf$groupnames = paste(sDf$agonistName, sDf$variable)
# to show also the data_summary of the antagonists we make a new dataframe: sDfFull, c
sDfFull = sDf
sDf = subset(sDf, variable == 'agonistEffect')
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
animalCount = list()
for (u in 1:length(unique(sDf$groupnames))){
    animalDf = subset(sDf, groupnames == unique(sDf$groupnames)[u])
    animalCount[u] = length(unique(animalDf$animalID))
    }
# to show also the data_summary of the antagonists we use sDfFull containing all the c
data_summaryFull = data_meanSDSEM(sDfFull, varname = "value", groupnames = "groupnames'
sumOut = capture.output(data_summaryFull)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = 
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dod
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position 
    scale_x_discrete(limits = xLabel)+
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = paste(sumRepOut))+
    theme(plot.caption = element_text(size = 8, hjust = 0))+
    theme(plot.title = element_text(size = 12))+
    geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
    geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
    theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank()
    theme(text=element_text(size=12))
```



```
           groupnames n    value       sd      sem
1    ACCDAMGO agonistEffect 7 92.75943 12.937997 4.890103
2 ACCDAMGO antagonistEffect 6 93.66419  7.969033 3.253344
3    ACCDPDPE agonistEffect 6 95.99529  6.288256 2.567170
4 ACCDPDPE antagonistEffect 5 86.01522  5.959387 2.665119
5     MDDAMGO agonistEffect 7 38.17238 16.075154 6.075837
6  MDDAMGO antagonistEffect 6 94.36899 10.609896 4.331472
7     MDDPDPE agonistEffect 6 99.57879 14.061706 5.740667
8  MDDPDPE antagonistEffect 5 92.05840 12.375464 5.534476
```

```
In [3]: # Perform statistic on subsetted data:
        trySDf =  melt(sDf, id.vars = c('cellID','baseName','agonistName','antagonistName'), me
        trySDf$source <- sapply(strsplit(as.character(trySDf$baseName),'BASE'),'[',2)
        trySDf$source <- sapply(strsplit(as.character(trySDf$source),'DAMGO'),'[',1)
        trySDf$source <- sapply(strsplit(as.character(trySDf$source),'DPDPE'),'[',1)
        trySDf$source = factor(trySDf$source, levels=unique(trySDf$source))
        for (i in 1:nrow(trySDf)){
            trySDf$opioidType[i] = as.character(trySDf$agonistName[i])
            if (trySDf$variable[i]=='baseValue'){
                trySDf$drug[i] = as.character(trySDf$baseName[i])
            }
            else if (trySDf$variable[i] == 'agonistValue'){
                trySDf$drug[i] = as.character(trySDf$agonistName[i])
            }
            else if (trySDf$variable[i] == 'antagonistValue'){
                trySDf$drug[i] = as.character(trySDf$antagonistName[i])
            }
            if (is.finite(trySDf$value[i])){
                if (trySDf$value[i]>=-1 & trySDf$value[i] <=0){ # for some unknown reason not 
                trySDf$value[i] = trySDf$value[i]*1000
            }}
        }

        trySDf$drug = gsub("MD","",trySDf$drug)
        trySDf$drug = gsub("ACC","",trySDf$drug)
        trySDf$drug = factor(trySDf$drug, levels=unique(trySDf$drug))
        trySDf$groupnames = paste(trySDf$drug, trySDf$source)
        trySDf$opioidType = gsub("MD","",trySDf$opioidType)
        trySDf$opioidType = gsub("ACC","",trySDf$opioidType)
        trySDf$opioidType = gsub("DAMGO","mu",trySDf$opioidType)
        trySDf$opioidType = gsub("DPDPE","delta",trySDf$opioidType)
        trySDf$opioidType = factor(trySDf$opioidType, levels=unique(trySDf$opioidType))
        ACCMD.df = data.frame(trySDf$cellID, trySDf$source, trySDf$drug, trySDf$opioidType, try
        colnames(ACCMD.df) = c('cellID', 'source', 'period', 'opioidType', 'amplitude','groupna
        ACCMD.df$period = gsub("BASEDPDPE","BASELINE",ACCMD.df$period)
        ACCMD.df$period = gsub("BASEDAMGO","BASELINE",ACCMD.df$period)
        ACCMD.df$period = gsub("DPDPENTD","ANTAGONIST",ACCMD.df$period)
        ACCMD.df$period = gsub("DAMGOCTAP","ANTAGONIST",ACCMD.df$period)
        ACCMD.df$period = gsub("DPDPE","AGONIST",ACCMD.df$period)
        ACCMD.df$period = gsub("DAMGO","AGONIST",ACCMD.df$period)

        ACCMD.df$groupnames = factor(ACCMD.df$groupnames,c('BASEDAMGO ACC','DAMGO ACC','DAMGOC
                                              'BASEDAMGO MD','DAMGO MD','DAMGOCTAP MD',
                                              'BASEDPDPE ACC','DPDPE ACC','DPDPENTD ACC',
                                              'BASEDPDPE MD','DPDPE MD','DPDPENTD MD'))
        ACCMD.df$period = factor (ACCMD.df$period,levels=c("BASELINE", "AGONIST", "ANTAGONIST"
        ACCMD.df$source = factor (ACCMD.df$source,levels=c("ACC","MD"))
        ACCMD.df$opioidType = factor (ACCMD.df$opioidType,levels=c("delta","mu"))
```

```
In [4]: library(lme4)
        library(multcomp)
        # We will be using linear combinatons of coefficients from the model to determine if d
        # When determining which coefficients to put in a linear combination, it's easiest to 
        # and subtract them. This leaves with the coefficient(s) they do not have in common.
        # For instance: baseline vs. agonist for the ACC/delta condition (baseline, ACC and de
        #   Baseline equation: frequency = intercept (since baseline, ACC and delta are the co
        #   Agonist equation:  frequency = intercept + periodAGONIST
        #   Difference in equations: periodAGONIST
        # Example 2: baseline vs. agonist for the MD/mu condition
        #   Baseline equation: frequency = intercept + sourceMD + opioidTypemu + sourceMD:opio
        #   Agonist equation : frequency = intercept + sourceMD + opioidTypemu + sourceMD:opio
        #   Difference in equations: periodAGONIST + periodAGONIST:opioidTypemu + periodAGONIS

        # To carry out these linear combinations first run the linear mixed effect model and s
        # This object is then used in the glht command (general linear hypothesis test - this 
        # In order to let R know which coefficients you would like to test in your linear comb
        # The matrix is based on the coefficients listed in the linear mixed effects regressio
        # are 12 coefficients total if you include the intercept.

        M0.ACC <- lmer(data= ACCMD.df, amplitude ~ period + source + opioidType + opioidType:pe
        summary(M0.ACC)
        ## Type III anova table with p-values for F-tests based on Satterthwaite's
        ## method:
        aov.ACC <- anova(M0.ACC, type="3")
        aov.ACC$Pvalue=pf(q=aov.ACC$F, df1 = aov.ACC$Df, df2 = length(unique(ACCMD.df$cellID))
        aov.ACC
```

```
Loading required package: Matrix
Loading required package: mvtnorm
Loading required package: survival
Loading required package: TH.data
Loading required package: MASS

Attaching package: TH.data

The following object is masked from package:MASS:

    geyser



Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID)
   Data: ACCMD.df

     AIC      BIC   logLik deviance df.resid
   965.5    997.8   -468.8    937.5       60

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.70742 -0.60419  0.02248  0.68216  1.98156

Random effects:
 Groups   Name        Variance Std.Dev.
 cellID   (Intercept)  6963     83.44
 Residual             15705    125.32
```

```
Number of obs: 74, groups:  cellID, 8

Fixed effects:
                                           Estimate Std. Error t value
(Intercept)                                -401.764     59.858  -6.712
periodAGONIST                                14.157     72.352   0.196
periodANTAGONIST                             22.092     76.426   0.289
sourceMD                                    -26.396     72.352  -0.365
opioidTypemu                                -16.502     71.067  -0.232
periodAGONIST:opioidTypemu                   16.555     98.599   0.168
periodANTAGONIST:opioidTypemu                12.193    103.702   0.118
sourceMD:opioidTypemu                       -44.920     98.599  -0.456
periodAGONIST:sourceMD:opioidTypedelta       -6.494    102.321  -0.063
periodANTAGONIST:sourceMD:opioidTypedelta    27.850    107.316   0.260
periodAGONIST:sourceMD:opioidTypemu         286.649     94.731   3.026
periodANTAGONIST:sourceMD:opioidTypemu        7.037     98.599   0.071

Correlation of Fixed Effects:
                           (Intr) prAGONIST prANTAGONIST sorcMD opdTyp
perdAGONIST                -0.604
prANTAGONIST               -0.575  0.473
sourceMD                   -0.604  0.500     0.473
opioidTypem                -0.643  0.509     0.496        0.509
prAGONIST:T                 0.443 -0.734    -0.347       -0.367 -0.694
pANTAGONIST:T               0.432 -0.349    -0.737       -0.349 -0.677
srcMD:pdTyp                 0.443 -0.367    -0.347       -0.734 -0.694
prdAGONIST:srcMD:pdTypd     0.427 -0.707    -0.335       -0.707 -0.360
prdANTAGONIST:srcMD:pdTypd  0.407 -0.337    -0.702       -0.674 -0.343
prdAGONIST:srcMD:pdTypm     0.000  0.000     0.000        0.000  0.333
prdANTAGONIST:srcMD:pdTypm  0.000  0.000     0.000        0.000  0.320
                           pAGONIST:T pANTAGONIST:T srMD:T
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
pANTAGONIST:T               0.475
srcMD:pdTyp                 0.500      0.475
prdAGONIST:srcMD:pdTypd     0.519      0.247         0.519
prdANTAGONIST:srcMD:pdTypd  0.247      0.517         0.495
prdAGONIST:srcMD:pdTypm    -0.480     -0.228        -0.480
prdANTAGONIST:srcMD:pdTypm -0.231     -0.475        -0.462
                           prdAGONIST:srcMD:pdTypd prdANTAGONIST:srcMD:pdTypd
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
```

```
pANTAGONIST:T
srcMD:pdTyp
prdAGONIST:srcMD:pdTypd
prdANTAGONIST:srcMD:pdTypd  0.477
prdAGONIST:srcMD:pdTypm     0.000                  0.000
prdANTAGONIST:srcMD:pdTypm  0.000                  0.000
                           prdAGONIST:srcMD:pdTypm
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
pANTAGONIST:T
srcMD:pdTyp
prdAGONIST:srcMD:pdTypd
prdANTAGONIST:srcMD:pdTypd
prdAGONIST:srcMD:pdTypm
prdANTAGONIST:srcMD:pdTypm  0.480
```

	Df	Sum Sq	Mean Sq	F value	Pvalue
period	2	129222.342	64611.171	4.11418128	0.05905561
source	1	1014.627	1014.627	0.06460742	0.80576780
opioidType	1	4765.247	4765.247	0.30343188	0.59678183
period:opioidType	2	109605.134	54802.567	3.48960856	0.08135860
source:opioidType	1	12242.165	12242.165	0.77953216	0.40303452
period:source:opioidType	4	184569.192	46142.298	2.93815721	0.09094361

```
In [5]: ##Linear combinations testing baseline vs antagonist for each source and Rx type
        ##This double checks assumption of experimental design; that after antagonist treatmen
        ##None of these comparisons are statistically significant, which is as expected
          ##baseline vs. antagonist, ACC and delta
            lc1a <- matrix(c(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0), 1)
            lincom1a.ACC <- glht(M0.ACC, linfct = lc1a)
            summary(lincom1a.ACC) ##This is the same as the coefficient periodANTAGONIST
          ##baseline vs. antagonist, ACC and mu
            lc1b <- matrix(c(0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0), 1)
            lincom1b.ACC <- glht(M0.ACC, linfct = lc1b)
            summary(lincom1b.ACC)
          ##baseline vs. antagonist, MD and delta
            lc1c <- matrix(c(0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0), 1)
            lincom1c.ACC <- glht(M0.ACC, linfct = lc1c)
            summary(lincom1c.ACC)
          ##baseline vs. antagonist, MD and mu
            lc1d <- matrix(c(0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1), 1)
            lincom1d.ACC <- glht(M0.ACC, linfct = lc1d)
            summary(lincom1d.ACC)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    22.09      76.43   0.289    0.773
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    34.28      70.15   0.489    0.625
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    49.94      76.43   0.653    0.513
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)
```

```
Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    41.32      70.15   0.589    0.556
(Adjusted p values reported -- single-step method)
```

```
In [6]: ##Linear combinations testing baseline vs agnoist for each source and Rx type
        ##This tests if the agonist had an effect on amplitude
        ##As expected, the only comparison that is statistically significant is for MD/mu
          ##baseline vs. agonist, ACC and delta
            lc2a <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), 1)
            lincom2a.ACC <- glht(M0.ACC, linfct = lc2a)
            summary(lincom2a.ACC) ##This is claim 3
          ##baseline vs. agonist, ACC and mu
            lc2b <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0), 1)
            lincom2b.ACC <- glht(M0.ACC, linfct = lc2b)
            summary(lincom2b.ACC)
          ##baseline vs. agonist, MD and delta
            lc2c <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0), 1)
            lincom2c.ACC <- glht(M0.ACC, linfct = lc2c)
            summary(lincom2c.ACC) ##Second part of claim 3
          ##baseline vs. agonist, MD and mu
            lc2d <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0), 1)
            lincom2d.ACC <- glht(M0.ACC, linfct = lc2d)
            summary(lincom2d.ACC) ##This is claim 2
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    14.16      72.35   0.196    0.845
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)
```

```
Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    30.71      66.99   0.459    0.647
(Adjusted p values reported -- single-step method)




 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0    7.664     72.352   0.106    0.916
(Adjusted p values reported -- single-step method)




 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = ACCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0   317.36      66.99   4.738 2.16e-06 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
(Adjusted p values reported -- single-step method)
```

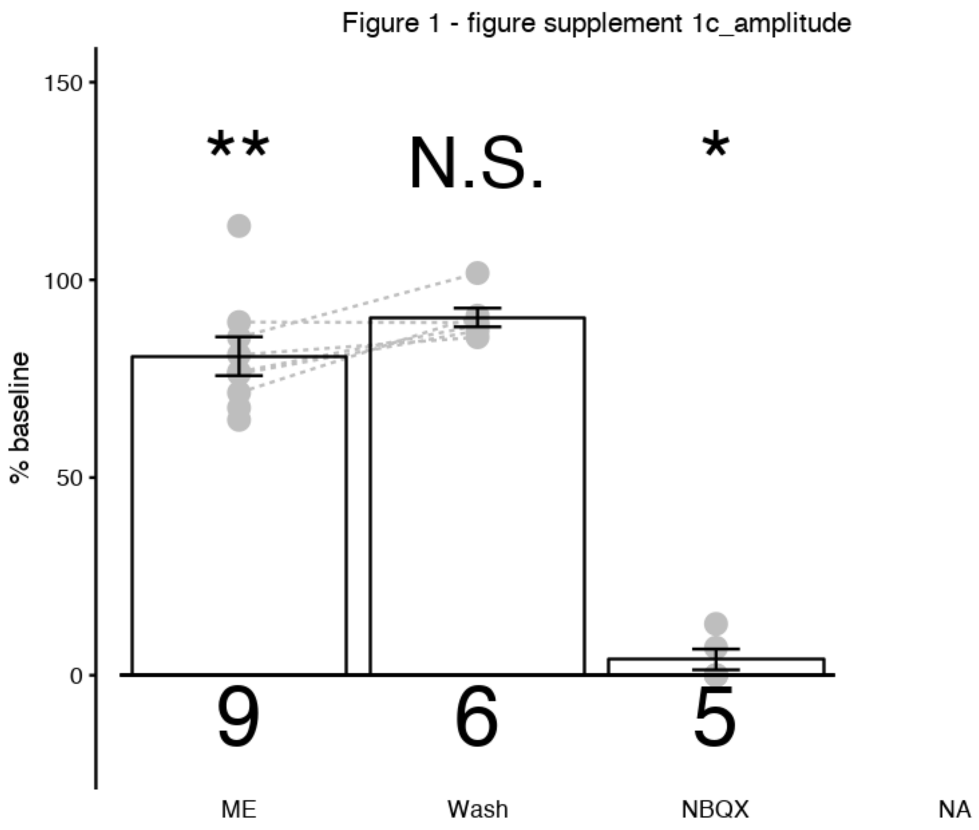
3 Figure 1 - figure supplement 1

3.1 Figure 1 - figure supplement 1c

```
In [7]: ###################################################################################
        #     Figure 1 - figure supplement 1c
        ###################################################################################
        # Subset: electric stim in DMS, recorded EPSC, from MSNs with Enk/Wash/NBQX
        tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & stimChannel == "E
        tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "ENK" | agonistName == "NB
```

```r
#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("ME", "Wash", "NBQX")
antagonistSelect = c("ENKWASH")
graphTitle = c("Figure 1 - figure supplement 1c")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# to avoid problems with sign reversal during agonist/antagonist treatment all conditi
# after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
    if (ssDf$baseValue[u]<0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]>0){
                ssDf$agonistValue[u]=-0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]>0){
                ssDf$antagonistValue[u]=-0.0001
            }
        }
    }
    if (ssDf$baseValue[u]>0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]<0){
                ssDf$agonistValue[u]=0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]<0){
                ssDf$antagonistValue[u]=0.0001
            }
        }
    }
    if (is.finite(ssDf$antagonistName[u])){
        if (ssDf$antagonistName[u] !=antagonistSelect){
            ssDf$antagonistValue[u] = NA
                                    }
    }
}
for (u in 1:nrow(sDf)){
    if (is.finite(sDf$antagonistName[u])){
            if(sDf$antagonistName[u] != antagonistSelect){
                sDf$antagonistValue[u]= NA
                if(sDf$variable[u] == 'antagonistEffect'){
                    sDf$value[u] = NA
                }
            }
        }
    }
    ssDf$baseValue = abs(ssDf$baseValue)
    ssDf$agonistValue =abs(ssDf$agonistValue)
    ssDf$antagonistValue = abs(ssDf$antagonistValue)

    #check if ssDf$antagonistValue contains only 'NA's
    checkNA = is.na(ssDf$antagonistValue)
    if (sum(checkNA)==length(checkNA)){
        ssDf$antagonistValue[checkNA] = 0
    }
    # In case of the NBQX there is no repeated measures (the experiments stand on themselv
    # are relative to baseline) So for the SM test we can only use the ENK data
    SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'ENK']))
    SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'ENK'][1:SMlen], ssDf$agonistValue[ssDf

    Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
    Bs = rep(1:SMlen,3)

    SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
    BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'ENK'][1:SMlen],ssDf$agonis
    BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'ENK'][1:SMlen],ssDf
    BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
    diff = c(ssDf$baseValue[ssDf$agonistName == 'ENK'][1:SMlen] - ssDf$agonistValue[ssDf$ag
    BAgWilcox$n = length(na.omit(diff[diff !=0]))

    BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'ENK'][1:SMlen],ssDf$antago
    BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'ENK'][1:SMlen],s
    BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
    diff = c(ssDf$baseValue[ssDf$agonistName == 'ENK'][1:SMlen] - ssDf$antagonistValue[ssD
    BAnWilcox$n = length(na.omit(diff[diff !=0]))

    AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'ENK'][1:SMlen], ssDf$an
    AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'ENK'][1:SMlen],
    AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
    diff = c(ssDf$agonistValue[ssDf$agonistName == 'ENK'][1:SMlen] - ssDf$antagonistValue[
    AgAnWilcox$n = length(na.omit(diff[diff !=0]))

    # we will check whether it is significant smaller when NBQX is added, since we already
    SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'NBQX']))
    BNBQXWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'NBQX'][1:SMlen],ssDf$ago
    BNBQXWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'NBQX'][1:SMlen],s
BNBQXWilcox$statistic = min(c(BNBQXWilcox$statistic, BNBQXWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'NBQX'][1:SMlen] - ssDf$agonistValue[ssDf$a
BNBQXWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-rank
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'ENK' & is.finite(s
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'ENK' & is.finit
antagonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'NBQX' & is.finite
NBQXAnimal = length(unique(animalDf$animalID))
reportString = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Numbe

k = 1
variableSelect = 'amplitude'
#since we adding more groups to the plot we need to make a plottingColumn to tell what
sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gr
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = N
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodg
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
    annotate("text", x=1, y=130, label = BAgAnno[1], size = as.numeric(BAgAnno[2]), fon
    annotate("text", x=2, y=130, label = BAnAnno[1], size =  as.numeric(BAnAnno[2]), fo
    annotate("text", x=3, y=130, label = BNBQXAnno[1], size =  as.numeric(BNBQXAnno[2])
    scale_x_discrete(labels = xLabel)+
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = paste(reportString,'\n',sumRepOut))+
    theme(plot.caption = element_text(size = 8, hjust = 0))+
    theme(plot.title = element_text(size = 12))+
    geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
    geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
            theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank()
            theme(text=element_text(size=12))
```

```
Warning message:
Removed 8 rows containing non-finite values (stat_summary).Warning message:
Removed 8 rows containing missing values (geom_point).Warning message:
Removed 8 rows containing missing values (geom_path).Warning message:
Removed 1 rows containing missing values (geom_bar).Warning message:
Removed 1 rows containing missing values (geom_errorbar).
```

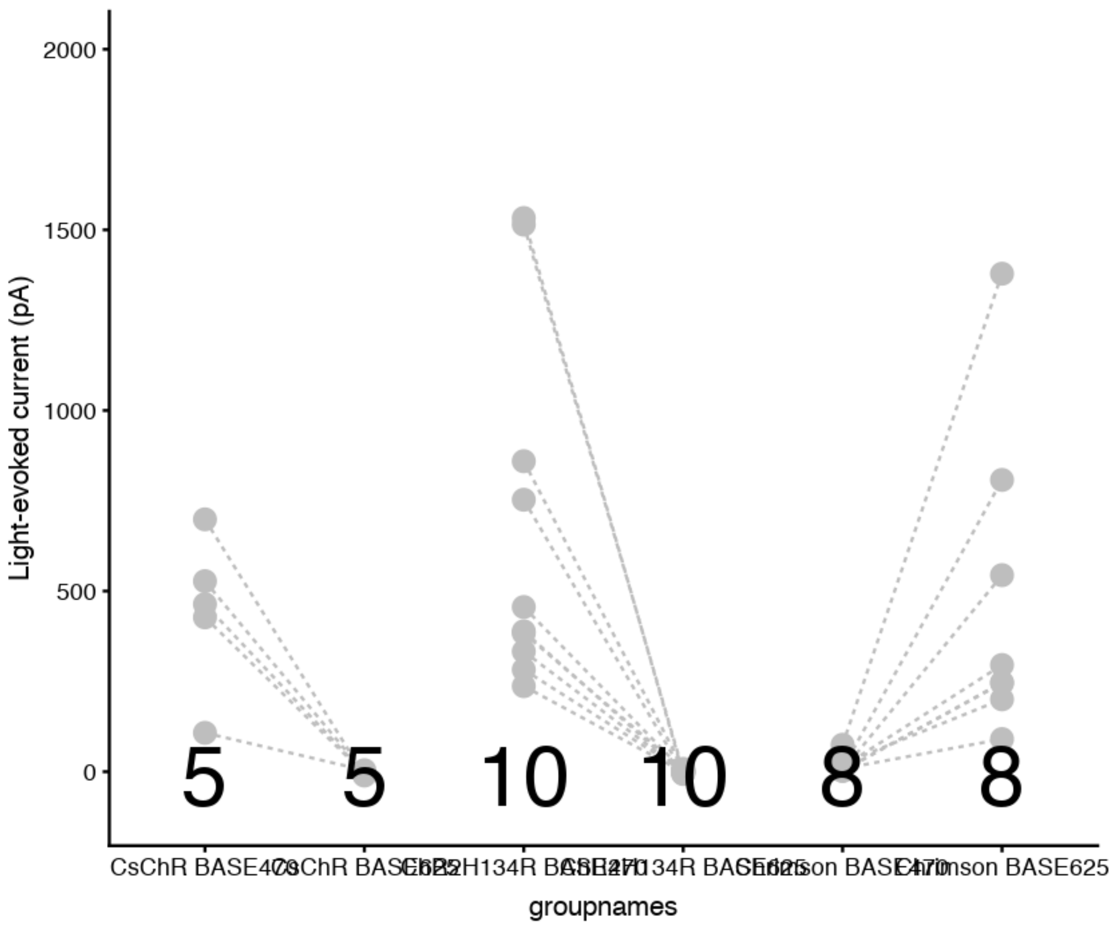


```
Skillings-Mack Statistic =  7.776709 , p-value =  0.020479
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  ME : Statistic: W( 9 )=  1  p-value =  0.0078125
 Baseline vs  Wash  : Statistic: W( 6 )=  1  p-value =  0.0625
 ME  vs  Wash  : Statistic: W( 6 )=  1  p-value =  0.0625
 Baseline vs  NBQX  : Statistic: W( 5 )=  0  p-value =  0.03125
 Number animals agonist:  5
 Number animals antagonist:  3
 Number animals NBQX:  4
            groupnames n       value          sd        sem
 1     ENK agonistEffect 9 80.684273712 14.721668276 4.907222759
 2  ENK antagonistEffect 6 90.491710119  5.807675520 2.370973602
 3    NBQX agonistEffect 5  3.992403393  5.863056072 2.622038387
 4 NBQX antagonistEffect 0         NaN          NA         NA
```

3.2 Figure 1 - figure supplement 1e

```
In [8]: ################################################################################
        #     Supplementary Figure 1d panel E
```

```
##################################################################################
#plot light-evoked currents for the different channels with 473 and 625 nm
chan = subset(df, baseName =="BASE470" & stimChannel !="ChR2H134R_Chrimson" & stimChann
chanDf = melt(chan, id.vars = c("cellID","animalID","stimChannel","baseName","agonistNa
chanDf$variable = gsub("baseValue","BASE470",chanDf$variable)
chanDf$value = chanDf$value*-1
chanDf$variable = gsub("agonistValue","BASE625",chanDf$variable)
chanDf$groupnames = paste(chanDf$stimChannel,chanDf$variable)

chanDf$groupnames = factor(chanDf$groupnames, levels = c("CsChR BASE470","CsChR BASE625

cschrTtest = t.test(chanDf$value[chanDf$groupnames=="CsChR BASE470"], chanDf$value[chan
chr2Ttest = t.test(chanDf$value[chanDf$groupnames=="ChR2H134R BASE470"], chanDf$value[c
chrimsonTtest = t.test(chanDf$value[chanDf$groupnames=="Chrimson BASE470"], chanDf$valu
cschrAnno = pvalAnno(cschrTtest$p.value,1)
chr2Anno = pvalAnno(chr2Ttest$p.value,1)
chrimsonAnno = pvalAnno(chrimsonTtest$p.value,1)
reportString1 = paste('Paired T-Tests: \n CsChR 470nm vs 625 nm: p-value = ',signif(cs
                     '\n ChR2(H134R) 470nm vs 625 nm: p-value = ',signif(chr2Ttest$p.va
                     '\n Chrimson 470nm vs 625 nm: p-value = ',signif(chrimsonTtest$p.v

animalDf = subset(chanDf, is.finite(chanDf$value) & groupnames == 'CsChR BASE470' | is
cschrAnimal = length(unique(animalDf$animalID))
animalDf = subset(chanDf, is.finite(chanDf$value) & groupnames == 'ChR2H134R BASE470'
chr2H134rAnimal = length(unique(animalDf$animalID))
animalDf = subset(chanDf, is.finite(chanDf$value) & groupnames == 'Chrimson BASE470' |
chrimsonAnimal = length(unique(animalDf$animalID))
reportString2 = paste(' Number animals CsChR: ',cschrAnimal,'\n Number animals ChR2(H1
data_summary = data_meanSDSEM(chanDf, varname = "value", groupnames = "groupnames")
sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}


ggplot(chanDf, aes(x = groupnames , y = value))+
  geom_point(data = chanDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
  geom_line(data = chanDf, aes(group = cellID ), lty = 2, colour = "gray")+
  coord_cartesian(ylim = c(-100,2000))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =chanDf, fun.data = give.n, geom = "text",fun.y = median, position
  labs(ylab('Light-evoked current (pA)'))+
  labs(caption = reportString)+
  labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  theme(text=element_text(size=12))
```



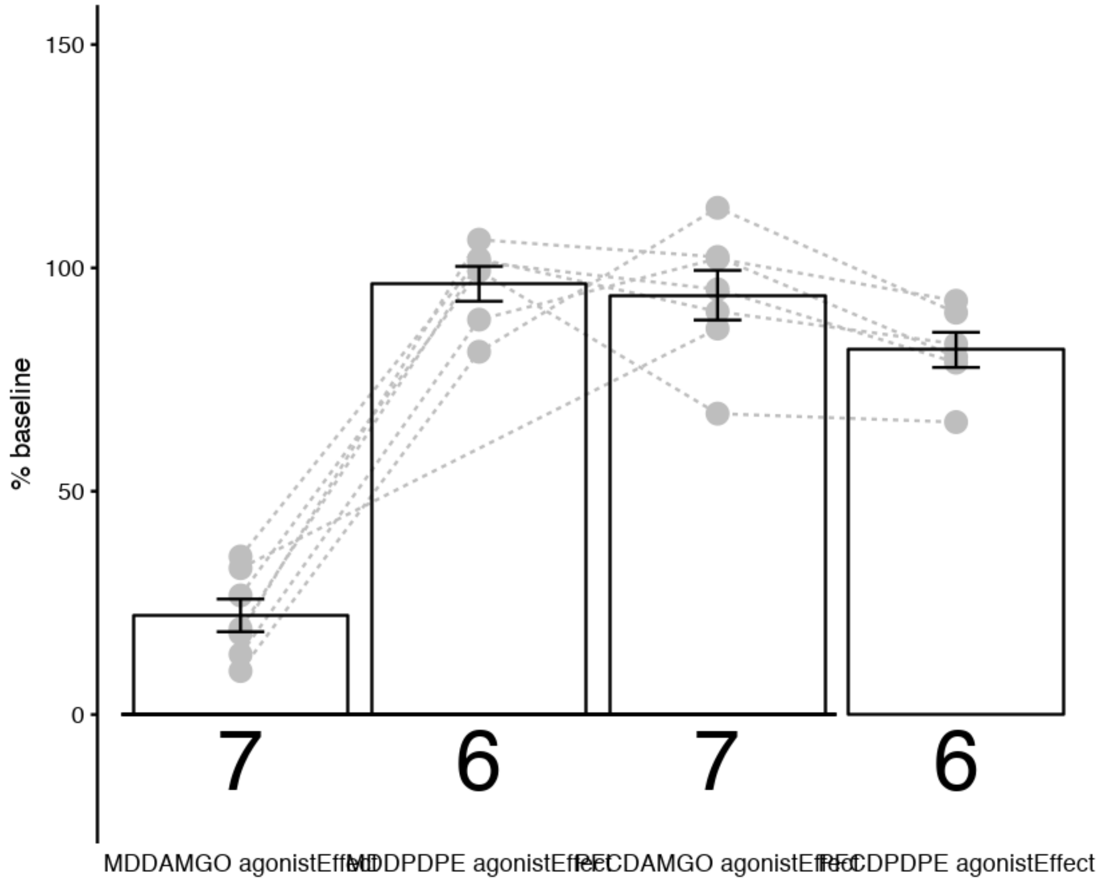
```
Paired T-Tests:
 CsChR 470nm vs 625 nm: p-value =  0.0103865
 ChR2(H134R) 470nm vs 625 nm: p-value =  0.00187509
 Chrimson 470nm vs 625 nm: p-value =  0.0159022
  Number animals CsChR:  2
 Number animals ChR2(H134R):  4
 Number animals Chrimson:  5
       groupnames  n      value          sd         sem
1    CsChR BASE470  5 444.8722000 215.431113265  96.343722746
2    CsChR BASE625  5  -4.0991940   5.280368660   2.361452654
3 ChR2H134R BASE470 10 673.9585000 489.733472108 154.867321829
4 ChR2H134R BASE625 10   0.1916075   4.373740141   1.383098074
5  Chrimson BASE470  8  16.8610300  24.229516901   8.566427853
6  Chrimson BASE625  8 476.0847750 429.691390916 151.918848167
```

3.3 Figure 1 - figure supplement 1h

```
In [9]: ###########################################################################################
        #     Supplementary Figure 1e panel C
        ###########################################################################################
        # Subset: two channel opto of PFC and MD wihtin DMS, recorded EPSC, from MSNs with DAM
        tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & stimSource == "MD
        tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "MDDAMGO" | agonistName ==

        #Important for good functioning of the script: No need to change:
        cols = colnames(tempSDF)
        tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("MDDAMGO agonistEffect", "MDDPDPE agonistEffect", "PFCDAMGO agonistEffect",
antagonistSelect = c("ENKWASH")
graphTitle = c("Figure 1 - figure supplement 1h")

k = 1
variableSelect = 'amplitude'
#since we adding more groups to the plot we need to make a plottingColumn to tell what
sDf$groupnames = paste(sDf$agonistName, sDf$variable)
# to show also the data_summary of the antagonists we use sDfFull containing all the c
sDfFull = sDf

sDf = subset(sDf, variable == 'agonistEffect')
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
animalCount = list()
for (u in 1:length(unique(sDf$groupnames))){
    animalDf = subset(sDf, groupnames == unique(sDf$groupnames)[u])
    animalCount[u] = length(unique(animalDf$animalID))
    }

# to show also the data_summary of the antagonists we use sDfFull containing all the c
data_summaryFull = data_meanSDSEM(sDfFull, varname = "value", groupnames = "groupnames
sumOut = capture.output(data_summaryFull)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill =
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dod
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position
    scale_x_discrete(limits = xLabel)+
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = sumRepOut)+
    theme(plot.caption = element_text(size = 8, hjust = 0))+
    theme(plot.title = element_text(size = 12))+
    geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
    geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
    theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank()
    theme(text=element_text(size=12))
```

Figure 1 - figure supplement 1h_amplitude



```
              groupnames n       value          sd         sem
1     MDDAMGO agonistEffect 7 22.20144044  9.680842743 3.659014625
2  MDDAMGO antagonistEffect 6 90.99290590  9.349664646 3.816984608
3     MDDPDPE agonistEffect 6 96.43610102  9.555016325 3.900819080
4  MDDPDPE antagonistEffect 5 85.26006450 10.107551431 4.520234417
5    PFCDAMGO agonistEffect 7 93.88351727 14.705309082 5.558084398
6 PFCDAMGO antagonistEffect 6 86.74916218 10.500669365 4.286880317
7    PFCDPDPE agonistEffect 6 81.64824047  9.635880182 3.933831612
8 PFCDPDPE antagonistEffect 5 87.07296536 16.708757875 7.472383685
```

```
In [10]: ## Similar to figure 1d, subset PFC/MD and perform LLM
         trySDf =  melt(sDf, id.vars = c('cellID','baseName','agonistName','antagonistName'), m
         trySDf$source <- sapply(strsplit(as.character(trySDf$baseName),'BASE'),'[',2)
         trySDf$source <- sapply(strsplit(as.character(trySDf$source),'DAMGO'),'[',1)
         trySDf$source <- sapply(strsplit(as.character(trySDf$source),'DPDPE'),'[',1)
         trySDf$source = factor(trySDf$source, levels=unique(trySDf$source))
         for (i in 1:nrow(trySDf)){
             trySDf$opioidType[i] = as.character(trySDf$agonistName[i])
             if (trySDf$variable[i]=='baseValue'){
                 trySDf$drug[i] = as.character(trySDf$baseName[i])
             }
             else if (trySDf$variable[i] == 'agonistValue'){
                 trySDf$drug[i] = as.character(trySDf$agonistName[i])
```

```
            }
            else if (trySDf$variable[i] == 'antagonistValue'){
                trySDf$drug[i] = as.character(trySDf$antagonistName[i])
            }
            if (is.finite(trySDf$value[i])){
                if (trySDf$value[i]>=-1 & trySDf$value[i] <=0){
                trySDf$value[i] = trySDf$value[i]*1000
            }}
        }
        trySDf$drug = gsub("MD","",trySDf$drug)
        trySDf$drug = gsub("PFC","",trySDf$drug)
        trySDf$drug = factor(trySDf$drug, levels=unique(trySDf$drug))
        trySDf$groupnames = paste(trySDf$drug, trySDf$source)
        trySDf$opioidType = gsub("MD","",trySDf$opioidType)
        trySDf$opioidType = gsub("PFC","",trySDf$opioidType)
        trySDf$opioidType = gsub("DAMGO","mu",trySDf$opioidType)
        trySDf$opioidType = gsub("DPDPE","delta",trySDf$opioidType)
        trySDf$opioidType = factor(trySDf$opioidType, levels=unique(trySDf$opioidType))
        PFCMD.df = data.frame(trySDf$cellID, trySDf$source, trySDf$drug, trySDf$opioidType, t
        colnames(PFCMD.df) = c('cellID', 'source', 'period', 'opioidType', 'amplitude','group
        PFCMD.df$period = gsub("BASEDPDPE","BASELINE",PFCMD.df$period)
        PFCMD.df$period = gsub("BASEDAMGO","BASELINE",PFCMD.df$period)
        PFCMD.df$period = gsub("DPDPENTD","ANTAGONIST",PFCMD.df$period)
        PFCMD.df$period = gsub("DAMGOCTAP","ANTAGONIST",PFCMD.df$period)
        PFCMD.df$period = gsub("DPDPE","AGONIST",PFCMD.df$period)
        PFCMD.df$period = gsub("DAMGO","AGONIST",PFCMD.df$period)

        PFCMD.df$groupnames = factor(PFCMD.df$groupnames,c('BASEDAMGO PFC','DAMGO PFC','DAMGO
                                       'BASEDAMGO MD','DAMGO MD','DAMGOCTAP MD',
                                       'BASEDPDPE PFC','DPDPE PFC','DPDPENTD PFC',
                                       'BASEDPDPE MD','DPDPE MD','DPDPENTD MD'))
        PFCMD.df$period = factor (PFCMD.df$period,levels=c("BASELINE", "AGONIST", "ANTAGONIST
        PFCMD.df$source = factor (PFCMD.df$source,levels=c("PFC","MD"))
        PFCMD.df$opioidType = factor (PFCMD.df$opioidType,levels=c("delta","mu"))
        M0.PFC <- lmer(data= PFCMD.df, amplitude ~ period + source + opioidType + opioidType:
        summary(M0.PFC)
        ## Type III anova table with p-values for F-tests based on Satterthwaite's
        ## method:
        aov.PFC <- anova(M0.PFC, type="3")
        aov.PFC$Pvalue=pf(q=aov.PFC$F, df1 = aov.PFC$Df, df2 = length(unique(PFCMD.df$cellID)
        aov.PFC

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID)
   Data: PFCMD.df

     AIC      BIC   logLik deviance df.resid
```

```
  1015.6   1047.8   -493.8    987.6       60

Scaled residuals:
        Min          1Q      Median          3Q         Max
-2.52346015 -0.62057228  0.05152351  0.67664902  1.85540157

Random effects:
 Groups   Name        Variance Std.Dev.
 cellID   (Intercept) 25398.50 159.3691
 Residual             29461.46 171.6434
Number of obs: 74, groups:  cellID, 7

Fixed effects:
                                            Estimate  Std. Error  t value
(Intercept)                              -437.826415   92.938000 -4.71095
periodAGONIST                              66.548367   99.098363  0.67154
periodANTAGONIST                           21.267651  104.361194  0.20379
sourceMD                                   82.446333   99.098363  0.83196
opioidTypemu                                1.411987   96.009863  0.01471
periodAGONIST:opioidTypemu                -61.745795  135.048305 -0.45721
periodANTAGONIST:opioidTypemu              27.419967  141.810325  0.19336
sourceMD:opioidTypemu                     -67.355476  135.048305 -0.49875
periodAGONIST:sourceMD:opioidTypedelta    -57.959033  140.146249 -0.41356
periodANTAGONIST:sourceMD:opioidTypedelta  26.047367  146.986626  0.17721
periodAGONIST:sourceMD:opioidTypemu       316.042357  129.750215  2.43578
periodANTAGONIST:sourceMD:opioidTypemu    -25.482690  135.048305 -0.18869

Correlation of Fixed Effects:
                           (Intr) prAGONIST prANTAGONIST sorcMD opdTyp
perdAGONIST                -0.533
prANTAGONIST               -0.506  0.475
sourceMD                   -0.533  0.500     0.475
opioidTypem                -0.561  0.516     0.490        0.516
prAGONIST:T                 0.391 -0.734    -0.348       -0.367 -0.703
pANTAGONIST:T               0.371 -0.349    -0.737       -0.349 -0.668
srcMD:pdTyp                 0.391 -0.367    -0.348       -0.734 -0.703
prdAGONIST:srcMD:pdTypd     0.377 -0.707    -0.336       -0.707 -0.365
prdANTAGONIST:srcMD:pdTypd  0.359 -0.337    -0.704       -0.674 -0.348
prdAGONIST:srcMD:pdTypm     0.000  0.000     0.000        0.000  0.338
prdANTAGONIST:srcMD:pdTypm  0.000  0.000     0.000        0.000  0.325
                           pAGONIST:T pANTAGONIST:T srMD:T
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
pANTAGONIST:T               0.476
srcMD:pdTyp                 0.500      0.476
```

```
prdAGONIST:srcMD:pdTypd     0.519      0.247         0.519
prdANTAGONIST:srcMD:pdTypd  0.247      0.518         0.495
prdAGONIST:srcMD:pdTypm    -0.480     -0.229        -0.480
prdANTAGONIST:srcMD:pdTypm -0.231     -0.476        -0.462
                           prdAGONIST:srcMD:pdTypd prdANTAGONIST:srcMD:pdTypd
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
pANTAGONIST:T
srcMD:pdTyp
prdAGONIST:srcMD:pdTypd
prdANTAGONIST:srcMD:pdTypd  0.477
prdAGONIST:srcMD:pdTypm     0.000                   0.000
prdANTAGONIST:srcMD:pdTypm  0.000                   0.000
                           prdAGONIST:srcMD:pdTypm
perdAGONIST
prANTAGONIST
sourceMD
opioidTypem
prAGONIST:T
pANTAGONIST:T
srcMD:pdTyp
prdAGONIST:srcMD:pdTypd
prdANTAGONIST:srcMD:pdTypd
prdAGONIST:srcMD:pdTypm
prdANTAGONIST:srcMD:pdTypm  0.480
```

	Df	Sum Sq	Mean Sq	F value	Pvalue
period	2	148098.912956	74049.456478	2.51343499097	0.15042117751
source	1	169852.237633	169852.237633	5.76523552860	0.04739016688
opioidType	1	2814.370919	2814.370919	0.09552721493	0.76626101068
period:opioidType	2	64604.571482	32302.285741	1.09642526943	0.38527199585
source:opioidType	1	10767.419582	10767.419582	0.36547478438	0.56455418868
period:source:opioidType	4	255891.886462	63972.971616	2.17141236388	0.17443974406

```
In [11]: ##Linear combinations testing baseline vs antagonist for each source and Rx type
         ##This double checks assumption of experimental design; that after antagonist treatme
         ##None of these comparisons are statistically significant, which is as expected
           ##baseline vs. antagonist, PFC and delta
             lc1a <- matrix(c(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0), 1)
             lincom1a.PFC <- glht(M0.PFC, linfct = lc1a)
             summary(lincom1a.PFC) ##This is the same as the coefficient periodANTAGONIST
           ##baseline vs. antagonist, PFC and mu
             lc1b <- matrix(c(0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0), 1)
             lincom1b.PFC <- glht(M0.PFC, linfct = lc1b)
             summary(lincom1b.PFC)
```

```
##baseline vs. antagonist, MD and delta
  lc1c <- matrix(c(0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0), 1)
  lincom1c.PFC <- glht(M0.PFC, linfct = lc1c)
  summary(lincom1c.PFC)
##baseline vs. antagonist, MD and mu
  lc1d <- matrix(c(0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1), 1)
  lincom1d.PFC <- glht(M0.PFC, linfct = lc1d)
  summary(lincom1d.PFC)
```

```

 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error z value Pr(>|z|)
1 == 0  21.26765  104.36119 0.20379  0.83852
(Adjusted p values reported -- single-step method)




 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0 48.68762   95.83388 0.50804  0.61142
(Adjusted p values reported -- single-step method)




 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error z value Pr(>|z|)
1 == 0  47.31502  104.36119 0.45338  0.65028
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0 23.20493   95.83388 0.24214  0.80867
(Adjusted p values reported -- single-step method)
```

```
In [12]: ##Linear combinations testing baseline vs agnoist for each source and Rx type
         ##This tests if the agonist had an effect on amplitude
         ##As expected, the only comparison that is statistically significant is for MD/mu
           ##baseline vs. agonist, PFC and delta
             lc2a <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0), 1)
             lincom2a.PFC <- glht(M0.PFC, linfct = lc2a)
             summary(lincom2a.PFC) ##This is claim 3
           ##baseline vs. agonist, PFC and mu
             lc2b <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0), 1)
             lincom2b.PFC <- glht(M0.PFC, linfct = lc2b)
             summary(lincom2b.PFC)
           ##baseline vs. agonist, MD and delta
             lc2c <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0), 1)
             lincom2c.PFC <- glht(M0.PFC, linfct = lc2c)
             summary(lincom2c.PFC) ##Second part of claim 3
           ##baseline vs. agonist, MD and mu
             lc2d <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0), 1)
             lincom2d.PFC <- glht(M0.PFC, linfct = lc2d)
             summary(lincom2d.PFC) ##This is claim 2
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
       Estimate Std. Error z value Pr(>|z|)
1 == 0 66.54837   99.09836 0.67154  0.50188
(Adjusted p values reported -- single-step method)
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error z value Pr(>|z|)
1 == 0  4.802571  91.747257 0.05235  0.95825
(Adjusted p values reported -- single-step method)



 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error z value Pr(>|z|)
1 == 0  8.589333  99.098363 0.08667  0.93093
(Adjusted p values reported -- single-step method)



 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error z value   Pr(>|z|)
1 == 0 320.84493   91.74726 3.49705 0.00047043 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
(Adjusted p values reported -- single-step method)
```

```
In [13]: ##Linear combination comparing the difference in baseline and agnoist for MD vs PFC w
           ##when rx=mu: (baseline vs. agonist for PFC) vs. (baseline vs. agonist for MD) - th
```

```
        lc3PFC <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0), 1)
        lc3MD  <- matrix(c(0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0), 1)
        lc3 <- lc3PFC - lc3MD
        lincom3.PFC <- glht(M0.PFC, linfct = lc3)
        summary(lincom3.PFC) ##This is claim 1
```

```
 Simultaneous Tests for General Linear Hypotheses

Fit: lmer(formula = amplitude ~ period + source + opioidType + opioidType:period +
    source:opioidType:period + source:opioidType + (1 | cellID),
    data = PFCMD.df, REML = FALSE)

Linear Hypotheses:
        Estimate Std. Error  z value Pr(>|z|)
1 == 0 -316.0424   129.7502 -2.43578  0.01486 *
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
(Adjusted p values reported -- single-step method)
```

4 Figure 1 - figure supplement 2

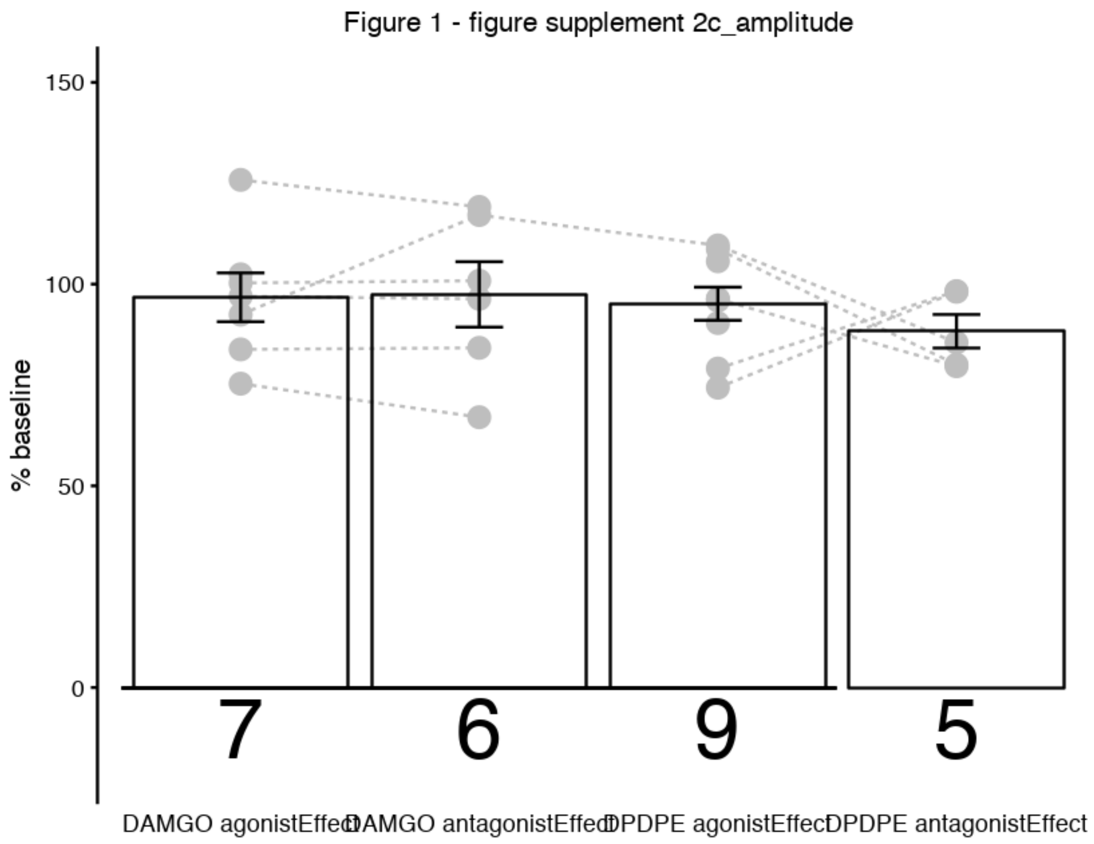
4.1 Figure 1 - figure supplement 2c

```
In [14]: ###########################################################################
         #     Figure 1 - figure supplement 2c
         ###########################################################################
         # Subset: opto stim ACC to DMS, recorded EPSC, from MSNs with DAMGO/DPDPE/antagonists
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & stimSource == "A
         tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" | agonistName ==

         #Important for good functioning of the script: No need to change:
         cols = colnames(tempSDF)
         tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

         # give the tempSDF a specific name, needs to be changed for every new subset
         sDf = subset(tempSDF, parameter == 'amplitude')
         xLabel = c("DAMGO", "NALOX", "DPDPE","NALTRI")
         antagonistSelect = c("DAMGONALOX","DPDPENALTRI")
         graphTitle = c("Figure 1 - figure supplement 2c")

         # prepare data into matrix for Skillings-Mack test.
         ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
         ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
         # # to avoid problems with sign reversal during agonist/antagonist treatment all cond
         # # after that the SMmatrix can be transformed to absolute values
```

```
for (u in 1:nrow(ssDf)){
    if (ssDf$baseValue[u]<0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]>0){
                ssDf$agonistValue[u]=-0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]>0){
                ssDf$antagonistValue[u]=-0.0001
            }
        }
    }
    if (ssDf$baseValue[u]>0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]<0){
                ssDf$agonistValue[u]=0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]<0){
                ssDf$antagonistValue[u]=0.0001
            }
        }
    }
    if (is.finite(ssDf$antagonistName[u])){
        if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=an
            ssDf$antagonistValue[u] = NA
                                    }
    }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru

# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
# Turns out that the statistic needs to be reported differently. One need the number

BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],s
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$ant
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMle
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistVal
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
# Turns out that the statistic needs to be reported differently. One need the number

BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$ago
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],s
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$ant
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
```

```
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = 
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dod
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position 
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
    theme(plot.caption = element_text(size = 5, hjust = 0))+
    theme(plot.title = element_text(size = 12))+
    geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
    geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
    theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank()
    theme(text=element_text(size=12))
```

```
Warning message:
Removed 7 rows containing non-finite values (stat_summary).Warning message:
Removed 7 rows containing missing values (geom_point).Warning message:
Removed 7 rows containing missing values (geom_path).
```



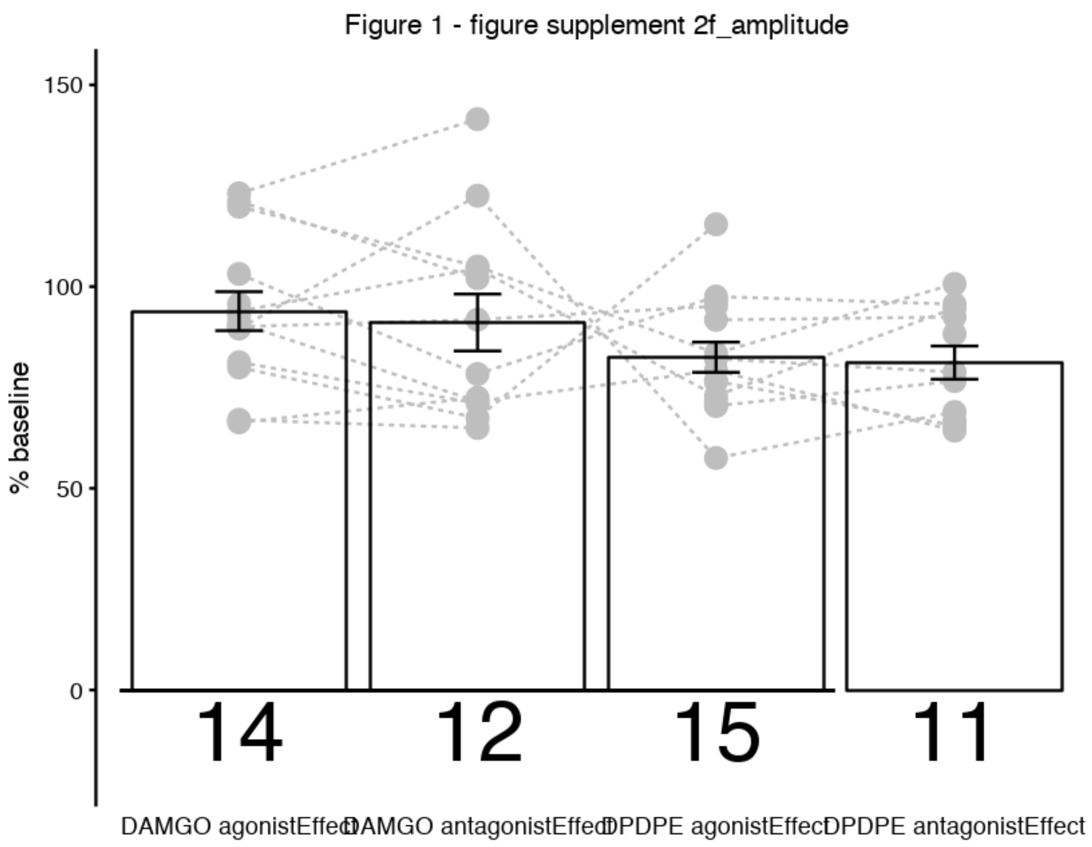
```
Skillings-Mack Statistic =  0.053590 , p-value =  0.973561
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 7 )=  11  p-value =  0.6875
 Baseline vs  NALOX  : Statistic: W( 6 )=  8  p-value =  0.6875
 DAMGO  vs  NALOX  : Statistic: W( 6 )=  10  p-value =  1

 Number animals agonist:  4
 Number animals antagonist:  4
 Skillings-Mack Statistic =  4.538461 , p-value =  0.103392
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 9 )=  18  p-value =  0.652344
 Baseline vs  NALTRI  : Statistic: W( 5 )=  0  p-value =  0.0625
 DPDPE  vs  NALTRI  : Statistic: W( 5 )=  6  p-value =  0.8125

 Number animals agonist:  5
 Number animals antagonist:  4
           groupnames n      value          sd        sem
 1    DAMGO agonistEffect 7 96.73004430 15.998971917 6.047042989
 2 DAMGO antagonistEffect 6 97.44623075 19.862503983 8.108833295
 3    DPDPE agonistEffect 9 95.13792834 12.347879066 4.115959689
 4 DPDPE antagonistEffect 5 88.31235597  9.305701805 4.161636363
```

4.2 Figure 1 - figure supplement 2f

```
In [15]: ##########################################################################################
         #     Figure 1 - figure supplement 2f
         ##########################################################################################
         # Subset: opto stim PFC to DMS, recorded EPSC, from MSNs with DAMGO/DPDPE/antagonists
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & stimSource == "PI
         tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" | agonistName == "
#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
# Since there are two antagonists used for
tempSDF$antagonistName=gsub("DAMGONALOX", "DAMGOCTAP",tempSDF$antagonistName)
tempSDF$antagonistName=gsub("DPDPENALTRI", "DPDPENALOX",tempSDF$antagonistName)

xLabel = c("DAMGO", "CTAP/NALOX", "DPDPE","NALOX/NAltri")
antagonistSelect = c("DAMGOCTAP","DPDPENALOX")
graphTitle = c("Figure 1 - figure supplement 2f")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antag
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

# DPDPE
##########
```

```
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))


BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
  coord_cartesian(ylim = c(-20,150))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
  labs(ylab('% baseline'))+
  labs(xlab(NULL))+
  labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
  labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
  geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
  theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())+
  theme(text=element_text(size=12))
```

```
Warning message in wilcox.test.default(ssDf$baseValue[ssDf$agonistName == "DPDPE"][1:SMlen], :
cannot compute exact p-value with tiesWarning message in wilcox.test.default(ssDf$agonistValue
cannot compute exact p-value with tiesWarning message:
Removed 6 rows containing non-finite values (stat_summary).Warning message:
Removed 6 rows containing missing values (geom_point).Warning message:
Removed 6 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  2.285714 , p-value =  0.318907
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 14 )=  38  p-value =  0.390991
 Baseline vs  CTAP/NALOX  : Statistic: W( 7 )=  8  p-value =  0.375
 DAMGO  vs  CTAP/NALOX  : Statistic: W( 7 )=  4  p-value =  0.109375

 Number animals agonist:  9
 Number animals antagonist:  8
 Skillings-Mack Statistic =  8.226922 , p-value =  0.016351
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 13 )=  4  p-value =  0.00415449
 Baseline vs  NALOX/NAltri  : Statistic: W( 2 )=  1  p-value =  1
 DPDPE  vs  NALOX/NAltri  : Statistic: W( 2 )=  1  p-value =  1

 Number animals agonist:  9
 Number animals antagonist:  6
            groupnames  n      value         sd        sem
 1    DAMGO agonistEffect 14 93.91726790 18.05255550 4.824748403
 2 DAMGO antagonistEffect 12 91.10507708 24.41810788 7.048900579
 3    DPDPE agonistEffect 15 82.49349471 14.51893673 3.748773345
 4 DPDPE antagonistEffect 11 81.15587185 13.64871654 4.115242874
```

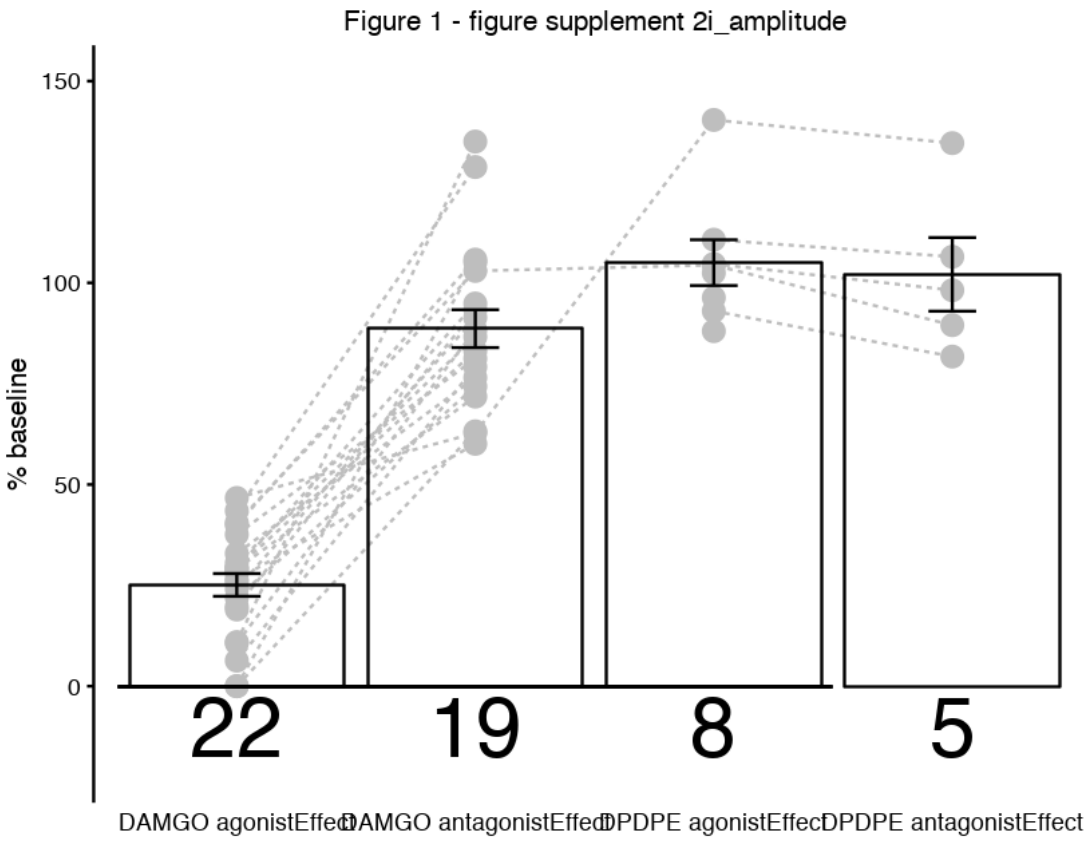
4.3 Figure 1 - figure supplement 2i

```
In [16]: #########################################################################
         #     Figure 1 - figure supplement 2i
         #########################################################################
         # Subset: opto stim in DMS from MD, recorded EPSC, from MSNs with Enk/Wash/NBQX
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
         tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" | agonistName == "

         #Important for good functioning of the script: No need to change:
         cols = colnames(tempSDF)
         tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("DAMGO", "NALOX", "DPDPE","NALTRI")
antagonistSelect = c("DAMGONALOX","DPDPENALTRI")
graphTitle = c("Figure 1 - figure supplement 2i")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
    if (ssDf$baseValue[u]<0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]>0){
                ssDf$agonistValue[u]=-0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]>0){
                ssDf$antagonistValue[u]=-0.0001
            }
        }
    }
    if (ssDf$baseValue[u]>0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]<0){
                ssDf$agonistValue[u]=0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]<0){
                ssDf$antagonistValue[u]=0.0001
            }
        }
    }
    if (is.finite(ssDf$antagonistName[u])){
        if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=an
            ssDf$antagonistValue[u] = NA
                                    }
    }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
```

```
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num
sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill =
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_doc
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
    theme(plot.caption = element_text(size = 5, hjust = 0))+
    theme(plot.title = element_text(size = 12))+
    geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
          geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
          theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank(
          theme(text=element_text(size=12))
```

```
Warning message:
Removed 6 rows containing non-finite values (stat_summary).Warning message:
Removed 6 rows containing missing values (geom_point).Warning message:
Removed 5 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  31.133330 , p-value =  0.000000
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 22 )=  0  p-value =  4.76837e-07
 Baseline vs  NALOX  : Statistic: W( 15 )=  37  p-value =  0.207764
 DAMGO  vs  NALOX  : Statistic: W( 15 )=  0  p-value =  6.10352e-05

 Number animals agonist:  16
 Number animals antagonist:  14
 Skillings-Mack Statistic =  2.521539 , p-value =  0.283436
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 8 )=  12  p-value =  0.460938
 Baseline vs  NALTRI  : Statistic: W( 4 )=  4  p-value =  0.875
 DPDPE  vs  NALTRI  : Statistic: W( 4 )=  0  p-value =  0.125

 Number animals agonist:  7
 Number animals antagonist:  5
           groupnames  n        value          sd         sem
 1    DAMGO agonistEffect 22  25.10390291 13.21712195 2.817899867
 2 DAMGO antagonistEffect 19  88.63857760 20.47014823 4.696174078
 3    DPDPE agonistEffect  8 104.99911937 16.02976276 5.667376973
 4 DPDPE antagonistEffect  5 102.09712472 20.41643154 9.130505755
```

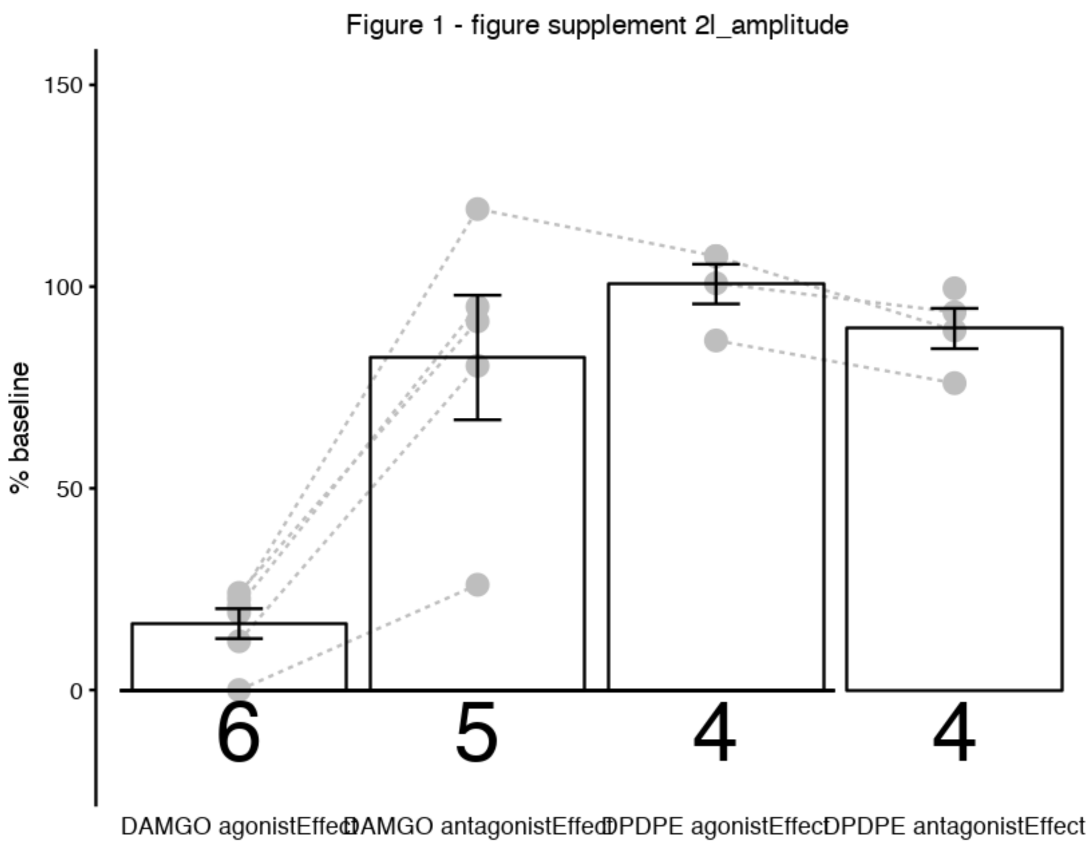
4.4 Figure 1 - figure supplement 2l

```
In [17]: ##################################################################################
         #     Figure 1 - figure supplement 2l
```

```
##############################################################################
# Subset: opto stim in DMS from AMthal, recorded EPSC, from MSNs with Enk/Wash/NBQX
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" | agonistName == "
#since there is a mixed bag of antagonist, lets rename the CTAP to NALOX and ICE to N
tempSDF$antagonistName = gsub("DAMGOCTAP","DAMGONALOX", tempSDF$antagonistName)
tempSDF$antagonistName = gsub("DAMGONLX","DAMGONALOX", tempSDF$antagonistName)
tempSDF$antagonistName = gsub("DPDPEICI","DPDPENALTRI", tempSDF$antagonistName)
tempSDF$antagonistName = gsub("DPDPENTD","DPDPENALTRI", tempSDF$antagonistName)
tempSDF$antagonistName = gsub("DPDPENLX","DPDPENALTRI", tempSDF$antagonistName)
tempSDF$antagonistName = as.factor(tempSDF$antagonistName)

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("DAMGO", "CTAP", "DPDPE","NALTRI")
antagonistSelect = c("DAMGONALOX","DPDPENALTRI")
graphTitle = c("Figure 1 - figure supplement 21")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
    if (ssDf$baseValue[u]<0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]>0){
                ssDf$agonistValue[u]=-0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]>0){
                ssDf$antagonistValue[u]=-0.0001
            }
        }
    }
    if (ssDf$baseValue[u]>0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]<0){
                ssDf$agonistValue[u]=0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]<0){
                ssDf$antagonistValue[u]=0.0001
            }
        }
    }
    if (is.finite(ssDf$antagonistName[u])){
        if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=an
            ssDf$antagonistValue[u] = NA
                                    }
    }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru

# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s

Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
```

```r
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s

Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
```

```
        sumOut = capture.output(data_summary)
        sumRepOut = sumOut[1]
        for(u in 2:length(sumOut)){
          sumRepOut = paste(sumRepOut,'\n', sumOut[u])
        }

        ggplot(data_summary, aes(x = groupnames , y = value))+
            geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "g
            geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
            geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill =
            geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_do
            coord_cartesian(ylim = c(-20,150))+
            theme_cowplot(font_size = 12)+
            stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position
            labs(ylab('% baseline'))+
            labs(xlab(NULL))+
            labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
            labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
            theme(plot.caption = element_text(size = 5, hjust = 0))+
            theme(plot.title = element_text(size = 12))+
            geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
            geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
            theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank(
            theme(text=element_text(size=12))

Warning message in wilcox.test.default(ssDf$baseValue[ssDf$agonistName == "DPDPE"][1:SMlen], :
cannot compute exact p-value with tiesWarning message in wilcox.test.default(ssDf$agonistValue
cannot compute exact p-value with tiesWarning message:
Removed 1 rows containing non-finite values (stat_summary).Warning message:
Removed 1 rows containing missing values (geom_point).
```



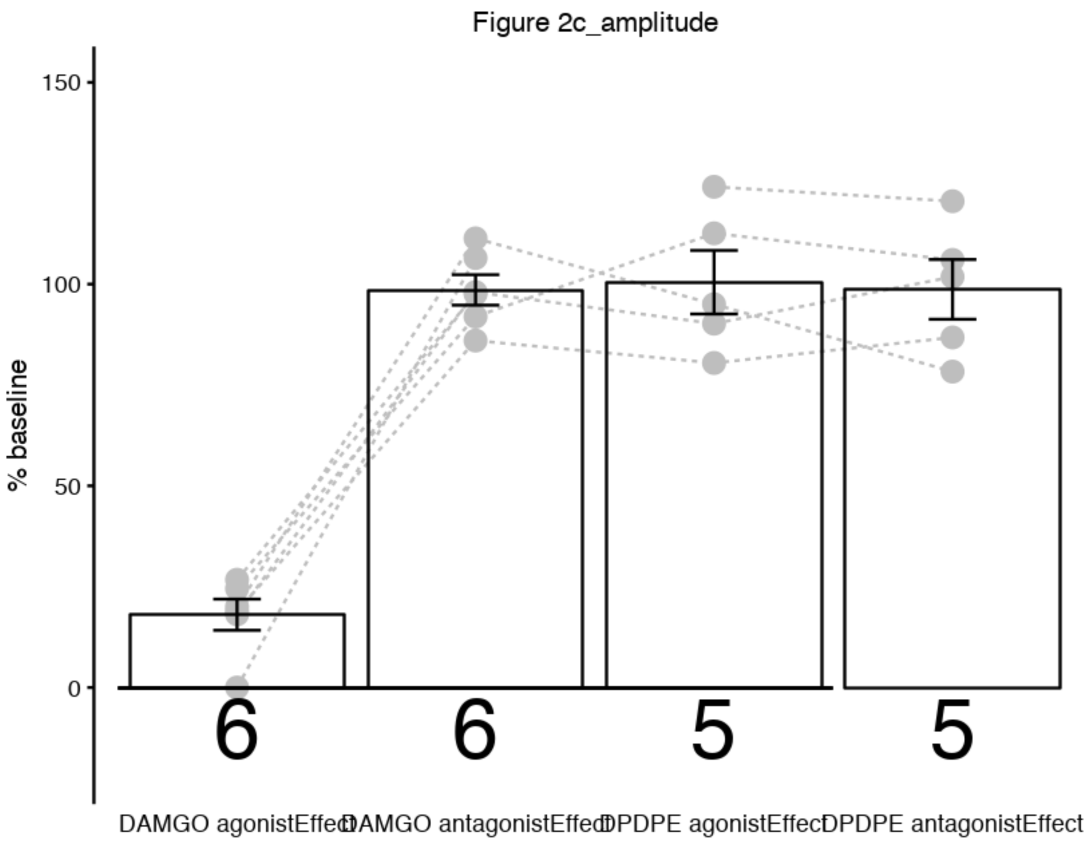
```
Skillings-Mack Statistic =  9.398642 , p-value =  0.009101
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 6 )=  0  p-value =  0.03125
 Baseline vs  CTAP  : Statistic: W( 5 )=  1  p-value =  0.125
 DAMGO  vs  CTAP  : Statistic: W( 5 )=  0  p-value =  0.0625

 Number animals agonist:  5
 Number animals antagonist:  4
 Skillings-Mack Statistic =  6.000000 , p-value =  0.049787
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 3 )=  0  p-value =  0.173568
 Baseline vs  NALTRI  : Statistic: W( 3 )=  0  p-value =  0.25
 DPDPE  vs  NALTRI  : Statistic: W( 3 )=  0  p-value =  0.25

 Number animals agonist:  2
 Number animals antagonist:  2
          groupnames n       value          sd         sem
 1    DAMGO agonistEffect 6  16.45943730  9.076301394  3.705384528
 2 DAMGO antagonistEffect 5  82.42432887 34.533092058 15.443668263
 3    DPDPE agonistEffect 4 100.64129281  9.852904983  4.926452491
 4 DPDPE antagonistEffect 4  89.61113911  9.986911897  4.993455948
```

5 Figure 2c

In [18]:
```
#############################################################################
#     Figure 2c
#############################################################################
# Subset: opto stim in DMS from vGLUT2cre positive cells MD, recorded EPSC, from MSNs
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType == "vGlu
tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" | agonistName == "

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```r
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("DAMGO", "CTAP", "DPDPE","NALTRI")
antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
graphTitle = c("Figure 2c")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
    if (ssDf$baseValue[u]<0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]>0){
                ssDf$agonistValue[u]=-0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]>0){
                ssDf$antagonistValue[u]=-0.0001
            }
        }
    }
    if (ssDf$baseValue[u]>0){
        if (is.finite(ssDf$agonistValue[u])){
            if (ssDf$agonistValue[u]<0){
                ssDf$agonistValue[u]=0.0001
            }
        }
        if (is.finite(ssDf$antagonistValue[u])){
            if (ssDf$antagonistValue[u]<0){
                ssDf$antagonistValue[u]=0.0001
            }
        }
    }
    if (is.finite(ssDf$antagonistName[u])){
        if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=a
            ssDf$antagonistValue[u] = NA
                                    }
    }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
```

```
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
BNBQXAnno = pvalAnno(BNBQXWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
    geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "
    geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
    geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill =
    geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_do
    coord_cartesian(ylim = c(-20,150))+
    theme_cowplot(font_size = 12)+
    stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position
    labs(ylab('% baseline'))+
    labs(xlab(NULL))+
    labs(title = paste(graphTitle,variableSelect[k],sep='_'))+
    labs(caption = paste(reportString1,"\n", reportString2,'\n',sumRepOut))+
    theme(plot.caption = element_text(size = 5, hjust = 0))+
theme(plot.title = element_text(size = 12))+
geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank(
theme(text=element_text(size=12))
```



```
Skillings-Mack Statistic =  9.333333 , p-value =  0.009404
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 6 )=  0  p-value =  0.03125
 Baseline vs  CTAP  : Statistic: W( 6 )=  7  p-value =  0.5625
 DAMGO  vs  CTAP  : Statistic: W( 6 )=  0  p-value =  0.03125

 Number animals agonist:  3
 Number animals antagonist:  3
 Skillings-Mack Statistic =  0.000000 , p-value =  1.000000
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 5 )=  7  p-value =  1
 Baseline vs  NALTRI  : Statistic: W( 5 )=  6  p-value =  0.8125
 DPDPE  vs  NALTRI  : Statistic: W( 5 )=  6  p-value =  0.8125

 Number animals agonist:  3
 Number animals antagonist:  3
          groupnames n       value          sd        sem
 1    DAMGO agonistEffect 6  18.05421877  9.461076336 3.862468240
 2 DAMGO antagonistEffect 6  98.55824916  9.258614599 3.779813582
 3    DPDPE agonistEffect 5 100.47377438 17.621503615 7.880575990
 4 DPDPE antagonistEffect 5  98.69228556 16.566879907 7.408933930
```

6 Figure 4

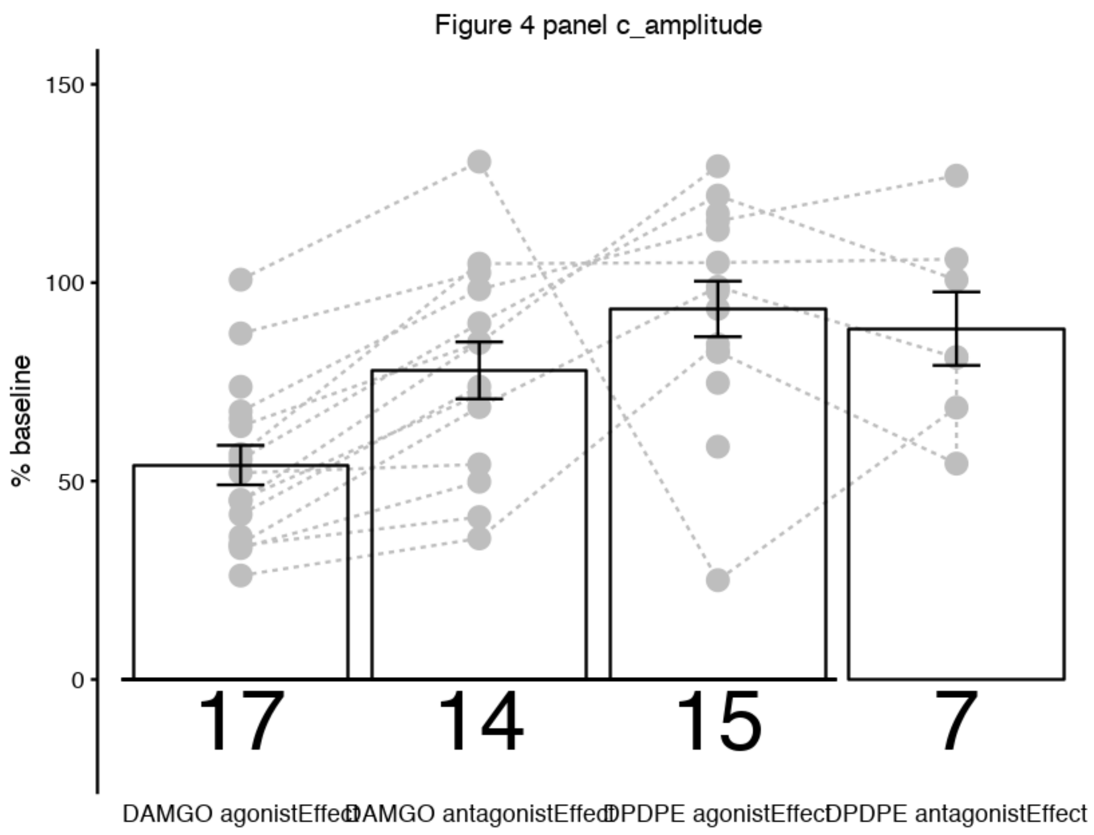
6.1 Figure 4c

```
In [19]: ###############################################################################
         #     Figure 4 panel c
         ###############################################################################
```

```
# Subset: photostim in ACC, recorded EPSC, from L5 PYR with DAMGO and DPDPE
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" |signal == "EPSC"

tempSDF$antagonistName = as.factor(tempSDF$antagonistName)

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude')
xLabel = c("DAMGO", "DAMGOCTAP", "DPDPE","NALTRI")
antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
graphTitle = c("Figure 4 panel c")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antago
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru

# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],s
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

# DPDPE
```

```
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodg
  coord_cartesian(ylim = c(-20,150))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
  labs(ylab('% baseline'))+
  labs(xlab(NULL))+
  labs(title = paste(graphTitle,"amplitude",sep='_'))+
  labs(caption = paste(reportString1,"\n\n", reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
  geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
  theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
  theme(text=element_text(size=12))
```

```
Warning message in wilcox.test.default(ssDf$baseValue[ssDf$agonistName == "DPDPE"][1:SMlen], :
cannot compute exact p-value with tiesWarning message in wilcox.test.default(ssDf$agonistValue
cannot compute exact p-value with tiesWarning message:
Removed 13 rows containing non-finite values (stat_summary).Warning message:
Removed 13 rows containing missing values (geom_point).Warning message:
Removed 13 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  22.850182 , p-value =  1.1e-05
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 17 )=  1  p-value =  3.05176e-05
 Baseline vs  DAMGOCTAP  : Statistic: W( 14 )=  16  p-value =  0.0202637
 DAMGO  vs  DAMGOCTAP  : Statistic: W( 14 )=  0  p-value =  0.00012207

 Number animals agonist:  13
 Number animals antagonist:  12

 Skillings-Mack Statistic =  0.988862 , p-value =  0.609918
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 14 )=  36  p-value =  0.315055
 Baseline vs  NALTRI  : Statistic: W( 4 )=  3  p-value =  0.625
 DPDPE  vs  NALTRI  : Statistic: W( 4 )=  1  p-value =  0.25

 Number animals agonist:  11
 Number animals antagonist:  5
          groupnames  n      value         sd        sem
 1    DAMGO agonistEffect 17 54.01942524 20.55069078 4.984274632
 2 DAMGO antagonistEffect 14 77.88663765 26.89580293 7.188205694
 3    DPDPE agonistEffect 15 93.38457567 27.12686189 7.004125623
 4 DPDPE antagonistEffect  7 88.41111678 24.50097319 9.260497422
```

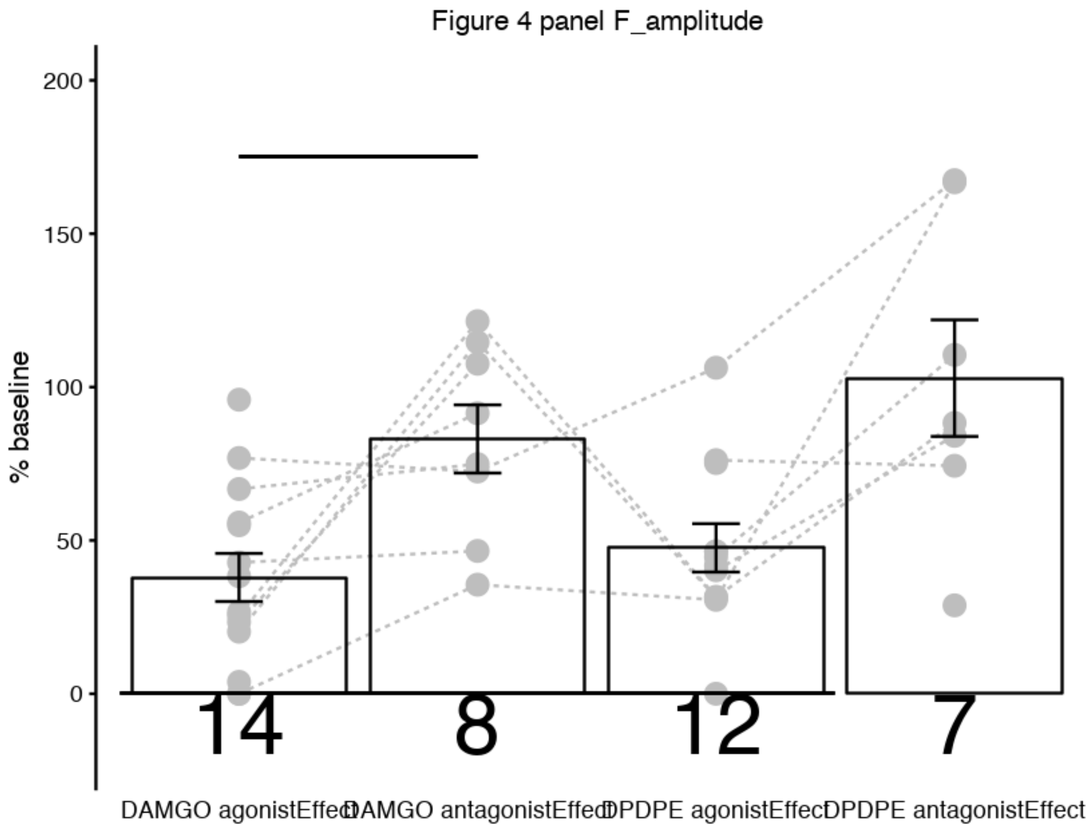
6.2 Figure 4f

In [20]:
```
#################################################################################
#     Figure 4 panel F
#################################################################################
# Subset: opto stim in ACC of MD input, recorded IPSC, to L5 Pyr
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
tempSDF = subset(tempSDF, signal == "IPSC" & agonistName == "DAMGO" |signal == "IPSC"

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude' )#& antagonistName != "DPDPEICI" & ant
sDf$value[sDf$antagonistName =="DPDPEICI" & sDf$variable =="antagonistEffect" ] = NA
sDf$value[sDf$antagonistName =="DPDPENALOX" & sDf$variable =="antagonistEffect" ] = N
xLabel = c("DAMGO", "DAMGOCTAP", "DPDPE","NALTRI")
antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
graphTitle = c("Figure 4 panel F")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antag
      ssDf$antagonistValue[u] = NA
    }
  }
}

ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)
```

```
# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$

AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num


sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
  coord_cartesian(ylim = c(-20,200))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
  labs(ylab('% baseline'))+
  labs(xlab(NULL))+
  labs(title = paste(graphTitle,"amplitude",sep='_'))+
  labs(caption = paste(reportString1,"\n\n", reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
  geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
  theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
  theme(text=element_text(size=12))
```

```
Warning message in wilcox.test.default(ssDf$baseValue[ssDf$agonistName == "DPDPE"][1:SMlen], :
cannot compute exact p-value with tiesWarning message in wilcox.test.default(ssDf$agonistValue
cannot compute exact p-value with tiesWarning message:
Removed 23 rows containing non-finite values (stat_summary).Warning message:
Removed 23 rows containing missing values (geom_point).Warning message:
Removed 22 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic = 15.678203 , p-value = 0.000394
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs DAMGO : Statistic: W( 14 )= 0 p-value = 0.00012207
 Baseline vs DAMGOCTAP : Statistic: W( 8 )= 10 p-value = 0.3125
 DAMGO vs DAMGOCTAP : Statistic: W( 8 )= 3 p-value = 0.0390625

 Number animals agonist: 9
 Number animals antagonist: 7

 Skillings-Mack Statistic = 7.425823 , p-value = 0.024406
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs DPDPE : Statistic: W( 11 )= 1 p-value = 0.00507749
 Baseline vs NALTRI : Statistic: W( 6 )= 6 p-value = 0.4375
 DPDPE vs NALTRI : Statistic: W( 6 )= 4 p-value = 0.21875

 Number animals agonist: 6
 Number animals antagonist: 5
          groupnames  n      value        sd        sem
 1    DAMGO agonistEffect 14  37.92881050 29.32996840  7.838763780
 2 DAMGO antagonistEffect  8  83.08159536 31.42050087 11.108824616
 3    DPDPE agonistEffect 12  47.57349404 27.29501377  7.879391775
 4 DPDPE antagonistEffect  7 102.89773640 50.28777079 19.006990786
```

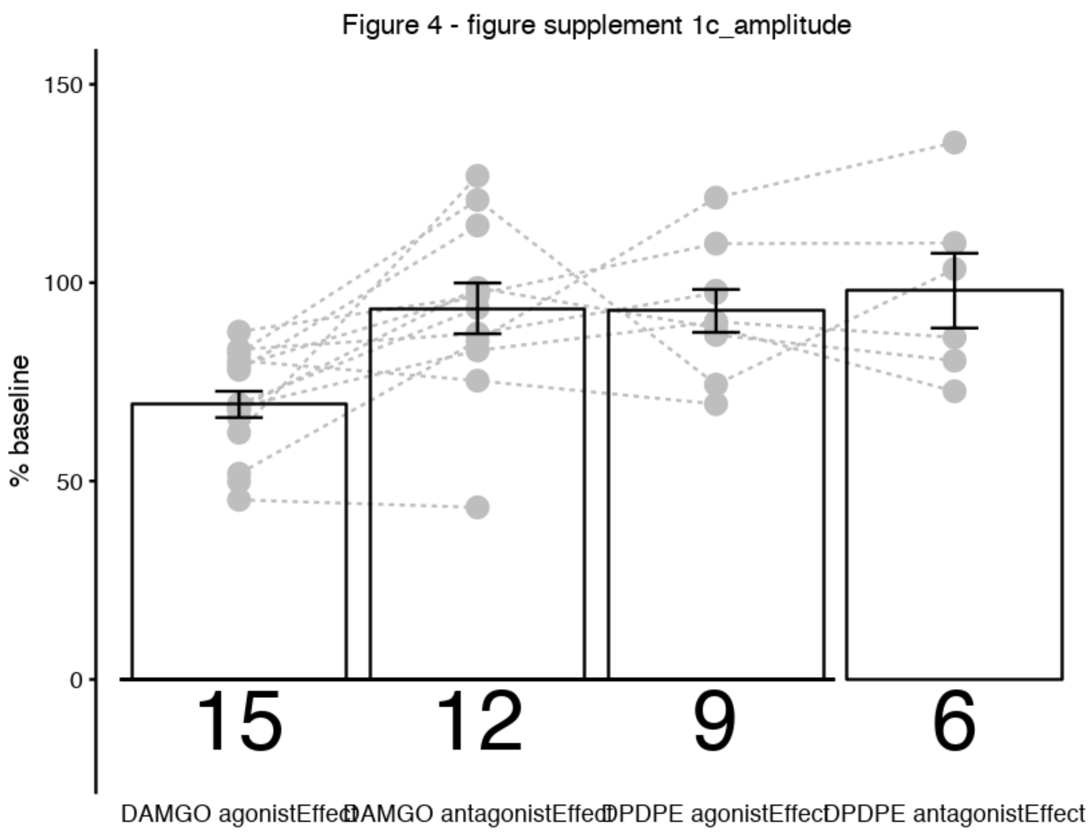
7 Figure 4 - figure supplement 1

7.1 Figure 4 - figure supplement 1c

```
In [21]: ####################################################################################
         #     Figure 4 - figure supplement 1c
         ####################################################################################
         # Subset: photostim in ACC, recorded EPSC, from L23 PYR with DAMGO and DPDPE
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
         tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DAMGO" |signal == "EPSC"

         tempSDF$antagonistName = as.factor(tempSDF$antagonistName)

         #Important for good functioning of the script: No need to change:
         cols = colnames(tempSDF)
         tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

         # give the tempSDF a specific name, needs to be changed for every new subset
         sDf = subset(tempSDF, parameter == 'amplitude')
         xLabel = c("DAMGO", "DAMGOCTAP", "DPDPE","NALTRI")
         antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
         graphTitle = c("Figure 4 - figure supplement 1c")

         # prepare data into matrix for Skillings-Mack test.
         ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
         ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
         # # to avoid problems with sign reversal during agonist/antagonist treatment all cond
         # # after that the SMmatrix can be transformed to absolute values
         for (u in 1:nrow(ssDf)){
           if (ssDf$baseValue[u]<0){
             if (is.finite(ssDf$agonistValue[u])){
               if (ssDf$agonistValue[u]>0){
                 ssDf$agonistValue[u]=-0.0001
               }
             }
             if (is.finite(ssDf$antagonistValue[u])){
               if (ssDf$antagonistValue[u]>0){
                 ssDf$antagonistValue[u]=-0.0001
               }
             }
           }
           if (ssDf$baseValue[u]>0){
             if (is.finite(ssDf$agonistValue[u])){
               if (ssDf$agonistValue[u]<0){
```

```
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antago
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$ago
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],s
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))


BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$ant
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[
BAnWilcox$n = length(na.omit(diff[diff !=0]))


AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMle
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistVal
AgAnWilcox$n = length(na.omit(diff[diff !=0]))
```

```
BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num
        sDf$groupnames = paste(sDf$agonistName, sDf$variable)
        data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

        sumOut = capture.output(data_summary)
        sumRepOut = sumOut[1]
        for(u in 2:length(sumOut)){
          sumRepOut = paste(sumRepOut,'\n', sumOut[u])
        }

        ggplot(data_summary, aes(x = groupnames , y = value))+
          geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
          geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
          geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
          geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
          coord_cartesian(ylim = c(-20,150))+
          theme_cowplot(font_size = 12)+
          stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
          labs(ylab('% baseline'))+
          labs(xlab(NULL))+
          labs(title = paste(graphTitle,"amplitude",sep='_'))+
          labs(caption = paste(reportString1,"\n\n", reportString2,'\n',sumRepOut))+
          theme(plot.caption = element_text(size = 5, hjust = 0))+
          theme(plot.title = element_text(size = 12))+
          geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
          geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
          theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
          theme(text=element_text(size=12))

Warning message:
Removed 6 rows containing non-finite values (stat_summary).Warning message:
Removed 6 rows containing missing values (geom_point).Warning message:
Removed 6 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic = 18.147709 , p-value = 0.000115
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs DAMGO : Statistic: W( 15 )= 0 p-value = 6.10352e-05
 Baseline vs DAMGOCTAP : Statistic: W( 12 )= 24 p-value = 0.266113
 DAMGO vs DAMGOCTAP : Statistic: W( 12 )= 5 p-value = 0.00488281

 Number animals agonist: 8
 Number animals antagonist: 8

 Skillings-Mack Statistic = 1.866025 , p-value = 0.393367
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs DPDPE : Statistic: W( 9 )= 9 p-value = 0.128906
 Baseline vs NALTRI : Statistic: W( 6 )= 8 p-value = 0.6875
 DPDPE vs NALTRI : Statistic: W( 6 )= 9 p-value = 0.84375

 Number animals agonist: 7
 Number animals antagonist: 6
          groupnames n     value        sd       sem
 1    DAMGO agonistEffect 15 69.31737792 12.84510342 3.316591442
 2 DAMGO antagonistEffect 12 93.52311737 22.19088380 6.405956366
 3    DPDPE agonistEffect 9 92.89156108 16.23144091 5.410480304
 4 DPDPE antagonistEffect 6 97.98803126 23.07060231 9.418533954
```

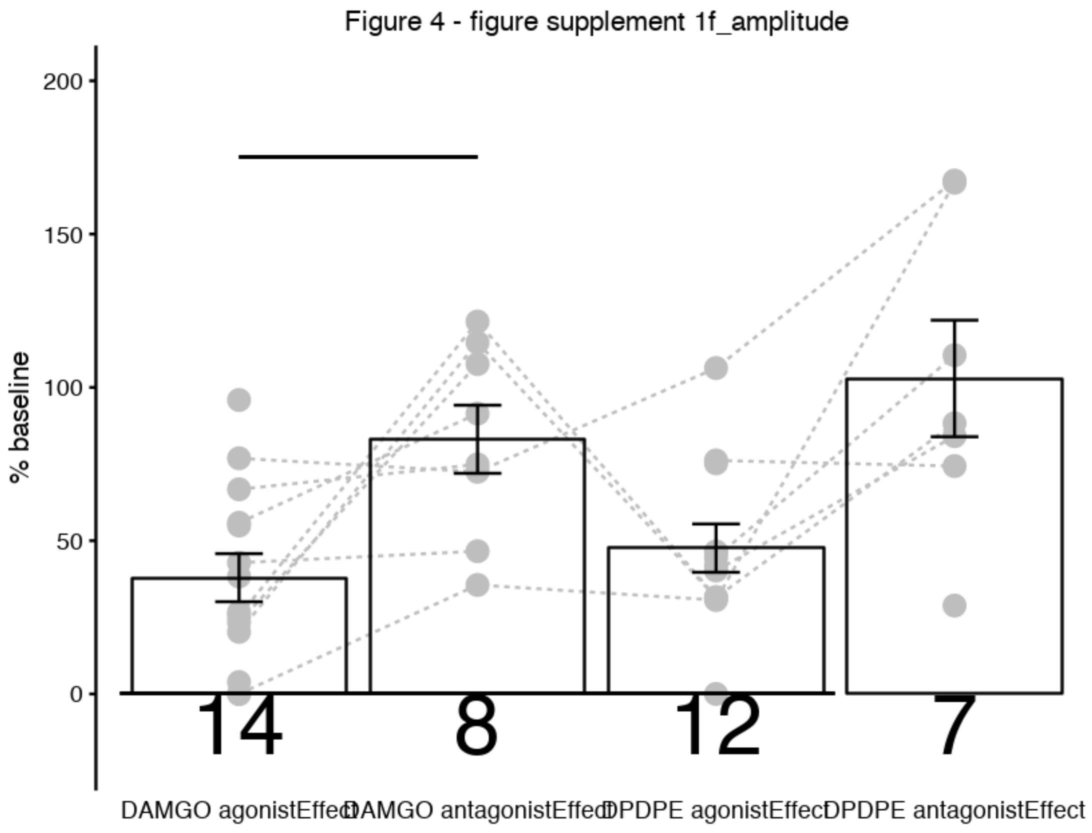
7.2 Figure 4 - figure supplement 1f

In [22]:
```
##########################################################################
#     Figure 4 - figure supplement 1f
##########################################################################
# Subset: opto stim in ACC of MD input, recorded IPSC, to L5 Pyr
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
tempSDF = subset(tempSDF, signal == "IPSC" & agonistName == "DAMGO" |signal == "IPSC"

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```r
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude' )#&& antagonistName != "DPDPEICI" & ant
sDf$value[sDf$antagonistName =="DPDPEICI" & sDf$variable =="antagonistEffect" ] = NA
sDf$value[sDf$antagonistName =="DPDPENALOX" & sDf$variable =="antagonistEffect" ] = N
xLabel = c("DAMGO", "DAMGOCTAP", "DPDPE","NALTRI")
antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
graphTitle = c("Figure 4 - figure supplement 1f")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antag
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)
```

```
# In this graph we will be having two sets of repeated measures omnibus test. Thus ru

# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
  coord_cartesian(ylim = c(-20,200))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
  labs(ylab('% baseline'))+
  labs(xlab(NULL))+
  labs(title = paste(graphTitle,"amplitude",sep='_'))+
  labs(caption = paste(reportString1,"\n\n", reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
  geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
  theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
  theme(text=element_text(size=12))
```

```
Warning message in wilcox.test.default(ssDf$baseValue[ssDf$agonistName == "DPDPE"][1:SMlen], :
cannot compute exact p-value with tiesWarning message in wilcox.test.default(ssDf$agonistValue
cannot compute exact p-value with tiesWarning message:
Removed 23 rows containing non-finite values (stat_summary).Warning message:
Removed 23 rows containing missing values (geom_point).Warning message:
Removed 22 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  15.678203 , p-value =  0.000394
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 14 )=  0  p-value =  0.00012207
 Baseline vs  DAMGOCTAP  : Statistic: W( 8 )=  10  p-value =  0.3125
 DAMGO  vs  DAMGOCTAP  : Statistic: W( 8 )=  3  p-value =  0.0390625

 Number animals agonist:  9
 Number animals antagonist:  7

 Skillings-Mack Statistic =  7.425823 , p-value =  0.024406
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 11 )=  1  p-value =  0.00507749
 Baseline vs  NALTRI  : Statistic: W( 6 )=  6  p-value =  0.4375
 DPDPE  vs  NALTRI  : Statistic: W( 6 )=  4  p-value =  0.21875

 Number animals agonist:  6
 Number animals antagonist:  5
           groupnames  n       value         sd          sem
 1    DAMGO agonistEffect 14  37.92881050 29.32996840  7.838763780
 2 DAMGO antagonistEffect  8  83.08159536 31.42050087 11.108824616
 3    DPDPE agonistEffect 12  47.57349404 27.29501377  7.879391775
 4 DPDPE antagonistEffect  7 102.89773640 50.28777079 19.006990786
```

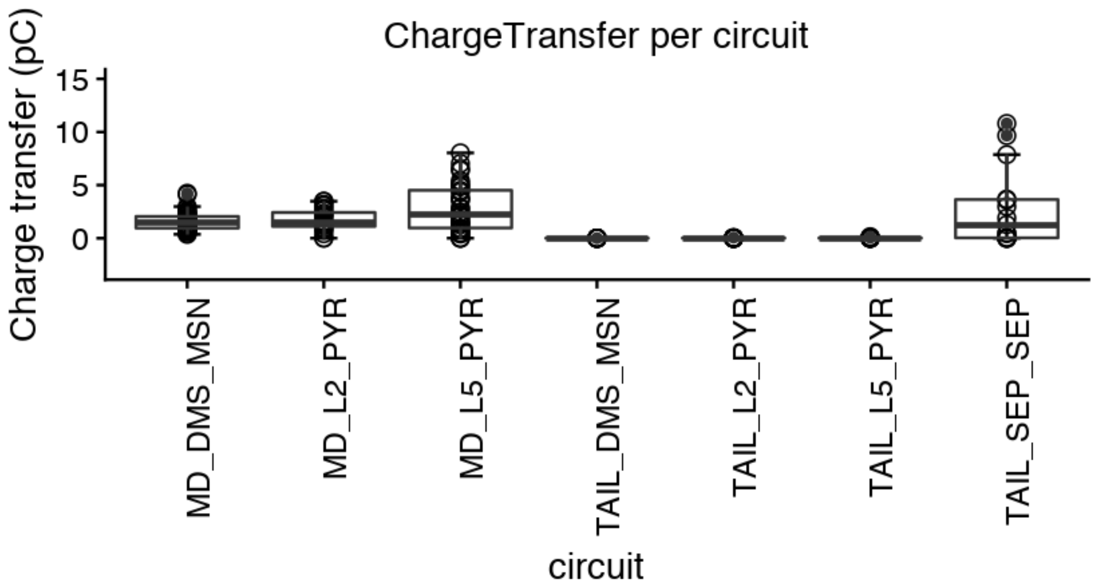
7.3 Figure 4 - figure supplement 2c

```
In [23]: ################################################################################
         #     Figure 4 - figure supplement 2c
         ################################################################################
         selectCell = read.csv("data/chargeTransferTailExp.csv" )
         selectCell$AUC = abs(selectCell$AUC)
         ############## FUNCTION ###############
         give.n <- function(x){
             return(c(y = -2, label = length(x)))
         }
         ######################################
         # Statistics for the tail injected only:
         tailDf = selectCell[grepl('TAIL',selectCell$circuit),]
         tailDf = subset(tailDf, !is.nan(AUC))
         r= capture.output(dunn.test(tailDf$AUC,tailDf$circuit, kw = TRUE, method = 'hochberg')
         reportString = paste(r[1],'\n',r[2],'\n',r[3],'\n',r[4],'\n',r[5],'\n',r[6],'\n',r[7]

         # Statistics for the all groups:
         r= capture.output(dunn.test(selectCell$AUC,selectCell$circuit, kw = TRUE, method = 'bo
         reportString = paste(r[1],'\n',r[2],'\n',r[3],'\n',r[4],'\n',r[5],'\n',r[6],'\n',r[7]

         # order selectCell factors:
         selectCell$circuit <- factor(selectCell$circuit, c('MD_DMS_MSN','MD_L2_PYR','MD_L5_PY

         # count number of cells tested;
         uniqAnimal= capture.output(aggregate(animalID ~ circuit,selectCell,function(x) length
         uniqCell= capture.output(aggregate(cellID ~ circuit,selectCell,function(x) length(uni

         #plot
         ggplot(selectCell, aes(x = circuit, y = AUC))+
             geom_point(shape = 1, colour = 'black', fill = NA, size = 3)+
             coord_cartesian(ylim = c(-3, 15))+
             stat_boxplot(geom = 'errorbar', width = 0.2)+
             geom_boxplot(varwidth = FALSE, notch = FALSE, fill = NA)+
             theme_cowplot(font_size = 16)+
             labs(ylab('Charge transfer (pC)'))+
             theme(axis.text.x=element_text(angle=90,hjust=1))+
             labs(title = 'ChargeTransfer per circuit')+
             labs(caption = reportString)+
             theme(plot.caption = element_text(size = 8, hjust = 0))+
             theme(plot.title = element_text(size = 16))

         #To count the amount of animals used:
         aggregate(animalID ~ circuit, selectCell, unique)
         reportString3 = paste(uniqAnimal[1],uniqAnimal[2],uniqAnimal[3],uniqAnimal[4],uniqAni
         #plot
         ggplot(selectCell, aes(x = circuit, y = AUC))+
           geom_point(shape = 1, colour = 'black', fill = NA, size = 3)+
           coord_cartesian(ylim = c(-3, 15))+
           stat_boxplot(geom = 'errorbar', width = 0.2)+
           geom_boxplot(varwidth = FALSE, notch = FALSE, fill = NA)+
           theme_cowplot(font_size = 16)+
           labs(ylab('Charge transfer (pC)'))+
           theme(axis.text.x=element_text(angle=90,hjust=1))+
           labs(title = 'ChargeTransfer per circuit')+
           labs(caption = reportString3)+
           theme(plot.caption = element_text(size = 8, hjust = 0))+
           theme(plot.title = element_text(size = 16))
```

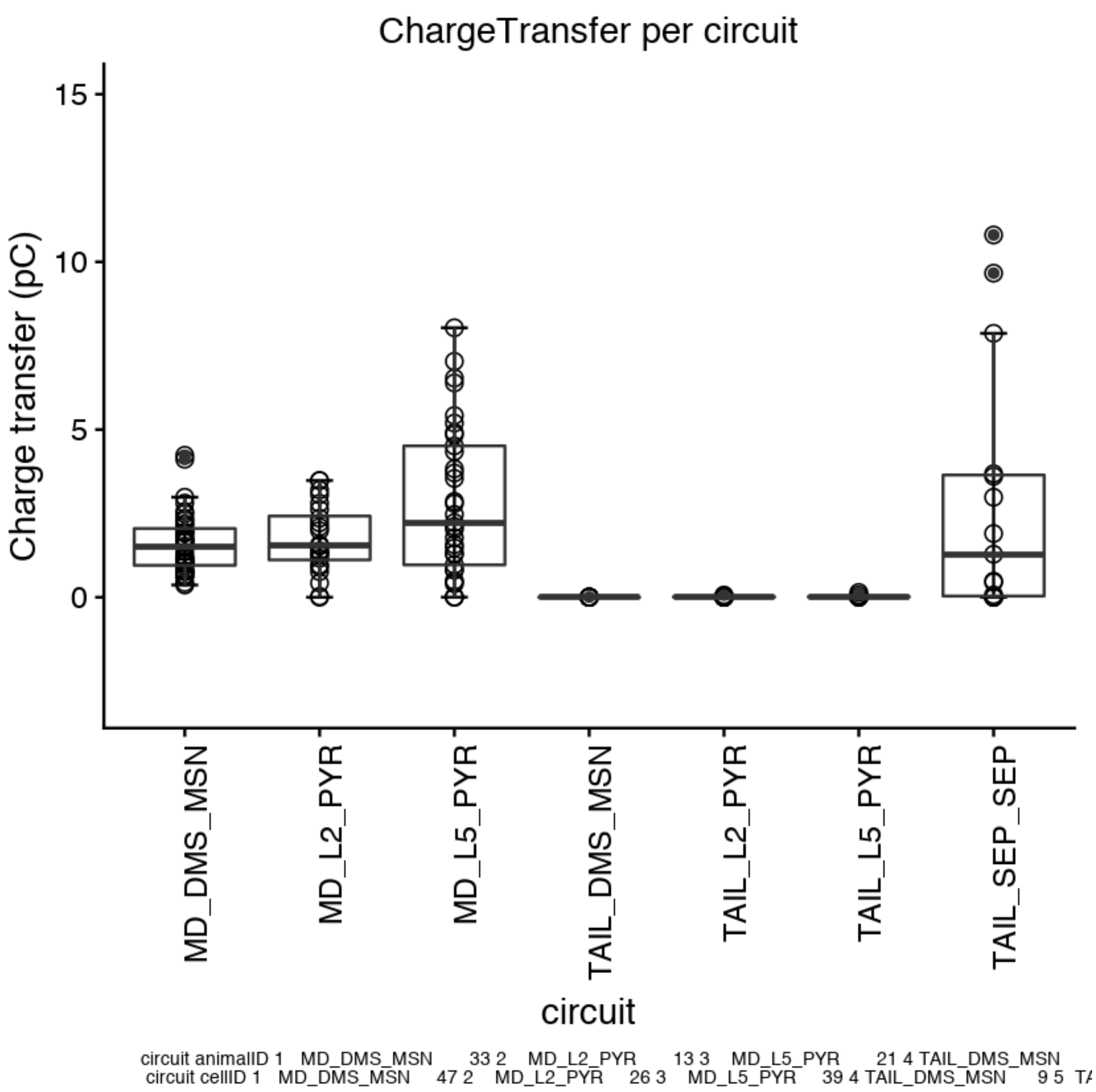
```
Warning message:
Removed 19 rows containing non-finite values (stat_boxplot).Warning message:
Removed 19 rows containing non-finite values (stat_boxplot).Warning message:
Removed 19 rows containing missing values (geom_point).
```

circuit	animalID
MD_DMS_MSN	43, 17, 12, 13, 14, 10, 11, 15, 16, 18, 19, 20, 27, 28, 7, 8, 9, 29, 32, 34, 36, 39, 50, 55, 56, 57, 45
MD_L2_PYR	2, 1, 3, 22, 23, 24, 25, 48, 50, 52, 53, 4, 49
MD_L5_PYR	2, 1, 23, 26, 6, 30, 31, 35, 33, 37, 38, 39, 40, 42, 43, 44, 47, 48, 21, 5, 49
TAIL_DMS_MSN	41, 51, 58
TAIL_L2_PYR	58, 59
TAIL_L5_PYR	41, 51, 58
TAIL_SEP_SEP	41, 51, 58, 59

```
Warning message:
Removed 19 rows containing non-finite values (stat_boxplot).Warning message:
Removed 19 rows containing non-finite values (stat_boxplot).Warning message:
Removed 19 rows containing missing values (geom_point).
```



```
 Kruskal-Wallis rank sum test

data: x and group
Kruskal-Wallis chi-squared = 69.6541, df = 6, p-value = 0


                       Comparison of x by group
                            (Bonferroni)
Col Mean-|
Row Mean |   MD_DMS_M   MD_L2_PY   MD_L5_PY   TAIL_DMS   TAIL_L2_   TAIL_L5_
---------+------------------------------------------------------------------
MD_L2_PY |  -0.187433
         |     1.0000
         |
MD_L5_PY |  -1.393359  -1.047211
         |     1.0000     1.0000
         |
TAIL_DMS |   4.760447   4.628266   5.557703
         |    0.0000*    0.0000*    0.0000*
         |
TAIL_L2_ |   4.713743   4.498887   5.657369  -0.778575
         |    0.0000*    0.0001*    0.0000*     1.0000
         |
TAIL_L5_ |   3.872337   3.753138   4.769718  -1.093314  -0.397188
         |    0.0011*    0.0018*    0.0000*     1.0000     1.0000
         |
```

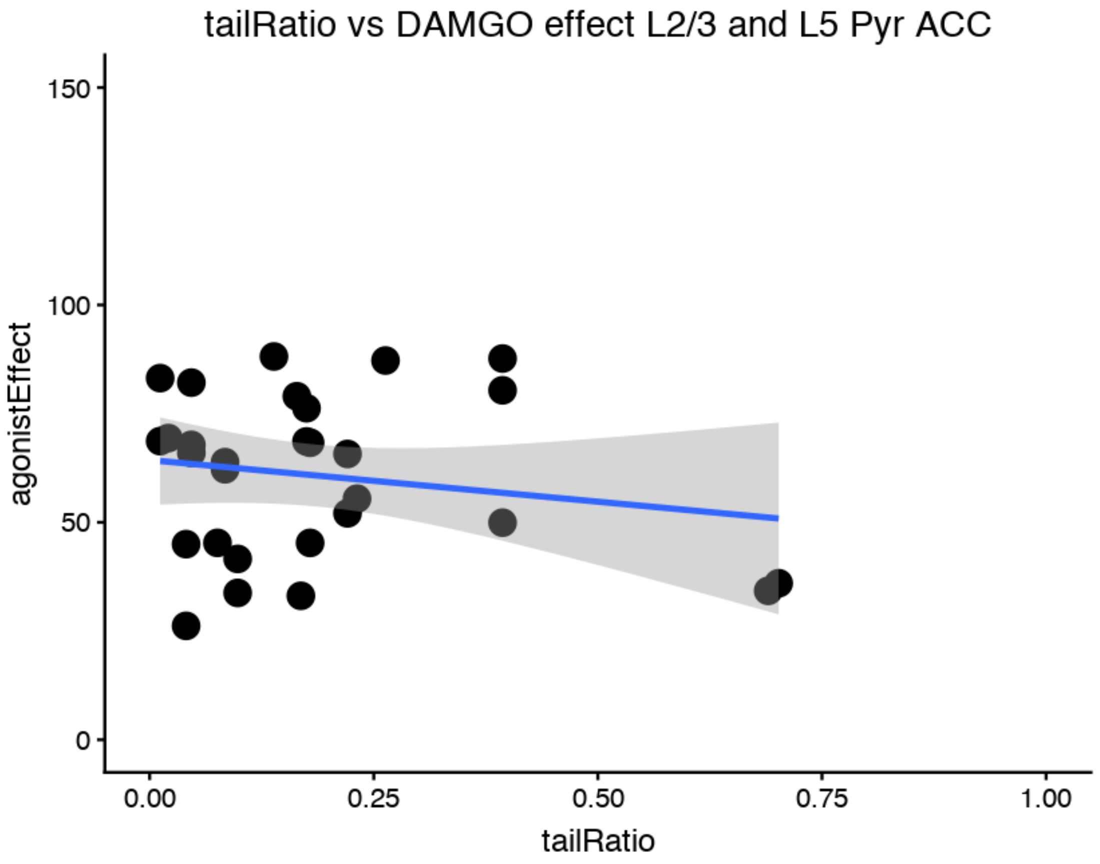


7.4 Figure 4 - figure supplement 2d

```
In [24]: ##############################################################################
         ##    Figure 4 - figure supplement 2d
         ##############################################################################
         # Subset all recordings of L2/3 and L5 pyramidal neurons in the ACC for which ENK was
         teDf = subset(df, signal == "EPSC" & parameter == "amplitude" & agonistName == "DAMGO
         teDf = subset(teDf, circuit =="MD_L2_PYR"|circuit == "MD_L5_PYR")

         #build linear regression model for tailRatio vs agonistEffect
         tail.model = lm(tailRatio~agonistEffect, data = teDf)
         modelOutput = capture.output(summary(tail.model))
         reportString = paste('Linear regression model (tailRatio ~ agonistEffect(Damgo)):\n',
```

```r
titleGr = 'tailRatio vs DAMGO effect L2/3 and L5 Pyr ACC'
# plot correlation graph
ggplot(teDf, aes(x=tailRatio, y=agonistEffect))+
  geom_point(size = 5)+
  geom_smooth(method=lm)+
  coord_cartesian(xlim = c(0, 1.0), ylim = c(0, 150))+
  labs(caption = reportString)+
  labs(title = titleGr)+
  theme(plot.caption = element_text(size = 16, hjust = 0))+
  theme(plot.title = element_text(size = 16))
```

```
Warning message:
Removed 6 rows containing non-finite values (stat_smooth).Warning message:
Removed 6 rows containing missing values (geom_point).
```



```
Linear regression model (tailRatio ~ agonistEffect(Damgo)):
                 Estimate    Std. Error  t value Pr(>|t|)
 (Intercept)     0.293184419  0.115197813  2.54505 0.016953 *
 agonistEffect -0.001763966  0.001814614 -0.97209 0.339635
 R-squared:  0.0338149179578441
 F-statistic: 0.9449564 on 1 and 27 DF,  p-value: 0.3396352
```

8 Figure 5

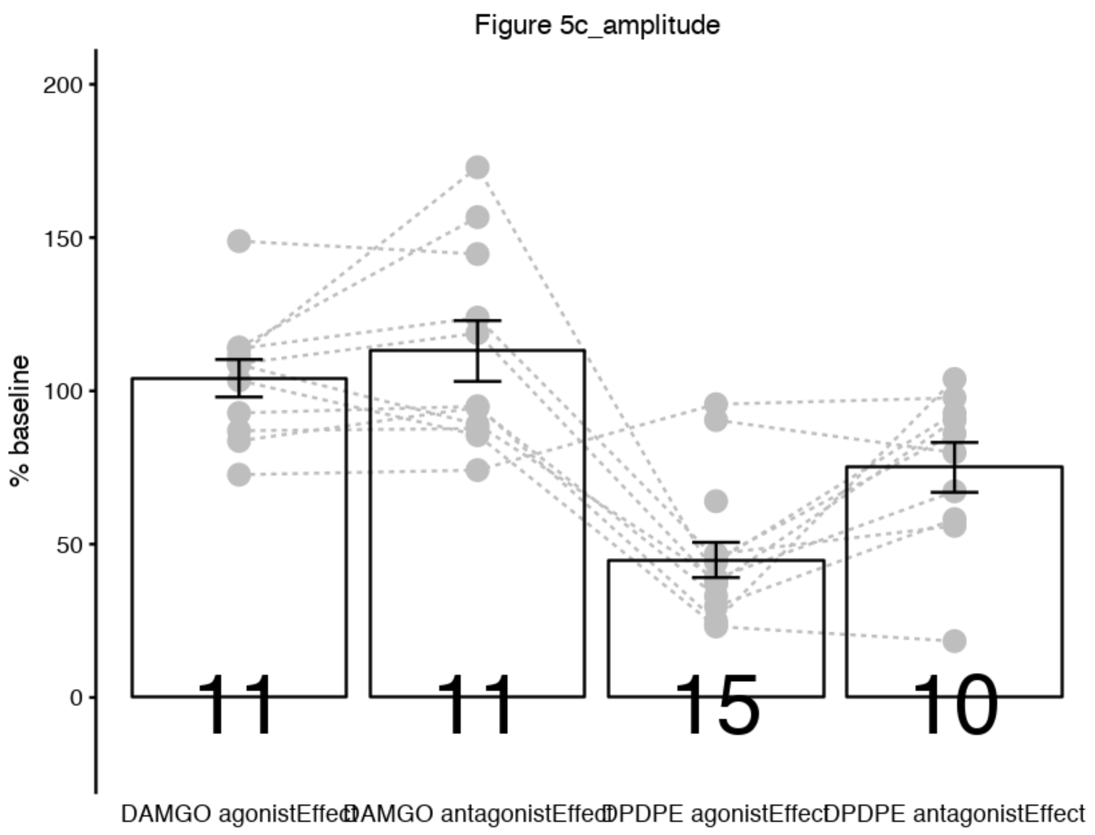
8.1 Figure 5c

```
In [25]: ########################################################################################
         #     Figure 5c
         ########################################################################################
         # Subset: opto stim of L5PV in ACC, recorded EPSC, from L5 PYR with DAMGO/DPDPE
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
         tempSDF = subset(tempSDF, signal == "IPSC" & agonistName == "DAMGO" |signal == "IPSC"

         #Important for good functioning of the script: No need to change:
         cols = colnames(tempSDF)
         tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

         # give the tempSDF a specific name, needs to be changed for every new subset
         sDf = subset(tempSDF, parameter == 'amplitude' )#& antagonistName != "DPDPEICI" & ant
         sDf$value[sDf$antagonistName =="DPDPEICI" & sDf$variable =="antagonistEffect" ] = NA
         sDf$value[sDf$antagonistName =="DPDPENALOX" & sDf$variable =="antagonistEffect" ] = N
         xLabel = c("DAMGO", "DAMGOCTAP", "DPDPE","NALTRI")
         antagonistSelect = c("DAMGOCTAP","DPDPENALTRI")
         graphTitle = c("Figure 5c")

         # prepare data into matrix for Skillings-Mack test.
         ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
         ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
         # # to avoid problems with sign reversal during agonist/antagonist treatment all cond
         # # after that the SMmatrix can be transformed to absolute values
         for (u in 1:nrow(ssDf)){
           if (ssDf$baseValue[u]<0){
             if (is.finite(ssDf$agonistValue[u])){
               if (ssDf$agonistValue[u]>0){
                 ssDf$agonistValue[u]=-0.0001
               }
             }
             if (is.finite(ssDf$antagonistValue[u])){
               if (ssDf$antagonistValue[u]>0){
                 ssDf$antagonistValue[u]=-0.0001
               }
             }
           }
           if (ssDf$baseValue[u]>0){
             if (is.finite(ssDf$agonistValue[u])){
               if (ssDf$agonistValue[u]<0){
                 ssDf$agonistValue[u]=0.0001
               }
             }
             if (is.finite(ssDf$antagonistValue[u])){
```

```
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1] & ssDf$antagonistName[u] !=antago
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# In this graph we will be having two sets of repeated measures omnibus test. Thus ru
# DAMGO
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DAMGO']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
# Turns out that the statistic needs to be reported differently. One need the number

BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DAMGO'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
```

```
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DAMGO' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DAMGO' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString1 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$ago
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],s
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssD
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$ant
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMle
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistVal
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu

sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")

sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
```

```
        for(u in 2:length(sumOut)){
          sumRepOut = paste(sumRepOut,'\n', sumOut[u])
        }

        ggplot(data_summary, aes(x = groupnames , y = value))+
          geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
          geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
          geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
          geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
          coord_cartesian(ylim = c(-20,200))+
          theme_cowplot(font_size = 12)+
          stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
          labs(ylab('% baseline'))+
          labs(xlab(NULL))+
          labs(title = paste(graphTitle,"amplitude",sep='_'))+
          labs(caption = paste(reportString1,"\n\n", reportString2,'\n',sumRepOut))+
          theme(plot.caption = element_text(size = 5, hjust = 0))+
          theme(plot.title = element_text(size = 12))+
          theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
          theme(text=element_text(size=12))

Warning message:
Removed 7 rows containing non-finite values (stat_summary).Warning message:
Removed 7 rows containing missing values (geom_point).Warning message:
Removed 7 rows containing missing values (geom_path).
```

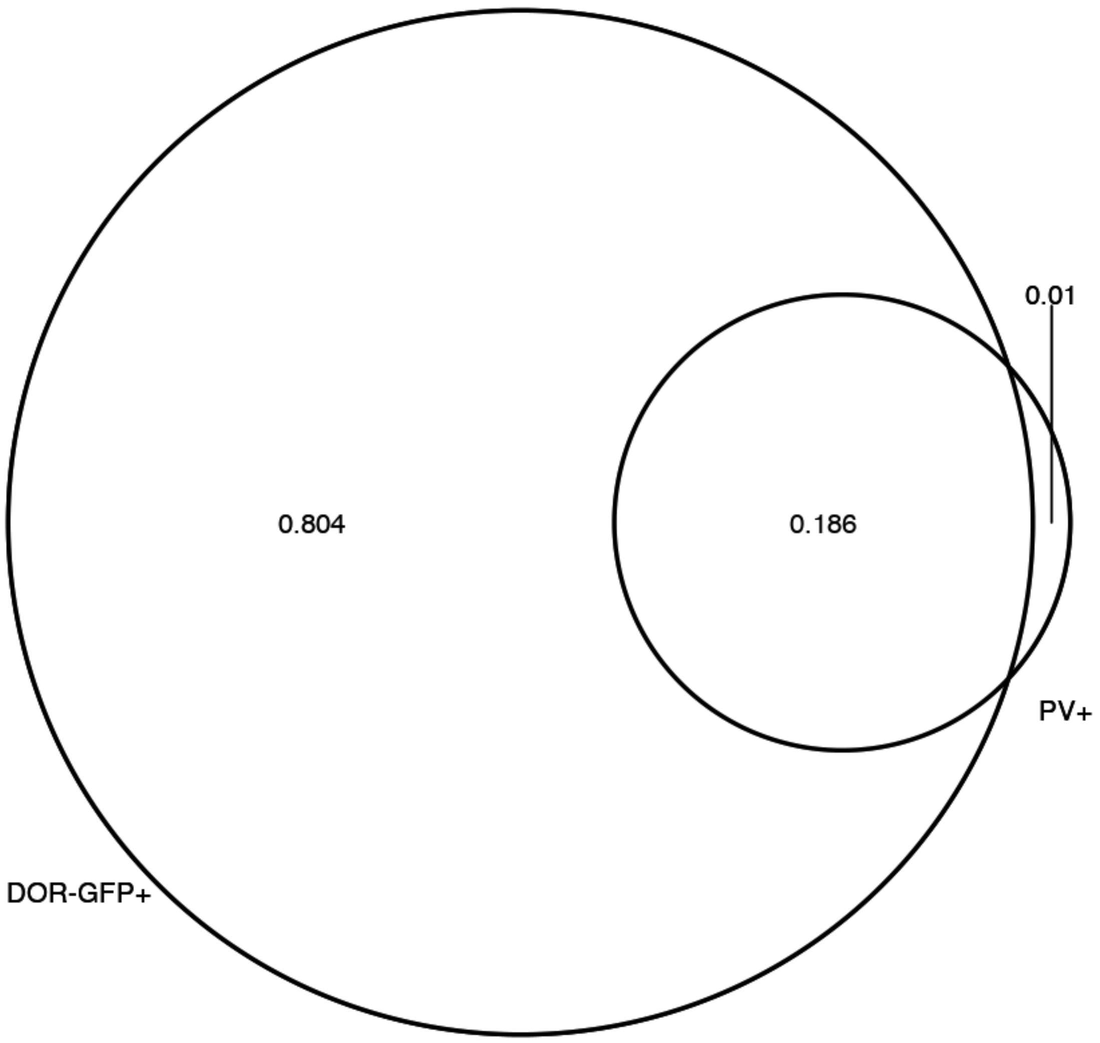


```
Skillings-Mack Statistic =  0.004632 , p-value =  0.997687
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DAMGO : Statistic: W( 11 )=  25  p-value =  0.519531
 Baseline vs  DAMGOCTAP  : Statistic: W( 9 )=  22  p-value = 1
 DAMGO  vs  DAMGOCTAP  : Statistic: W( 9 )=  18  p-value =  0.652344

 Number animals agonist:  5
 Number animals antagonist:  5

 Skillings-Mack Statistic =  19.598716 , p-value =  5.5e-05
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 15 )=  0  p-value =  6.10352e-05
 Baseline vs  NALTRI  : Statistic: W( 10 )=  2  p-value =  0.00585938
 DPDPE  vs  NALTRI  : Statistic: W( 10 )=  5  p-value =  0.0195312

 Number animals agonist:  8
 Number animals antagonist:  5
          groupnames  n        value          sd         sem
1    DAMGO agonistEffect 11 104.11967940 20.29828169 6.120162205
2 DAMGO antagonistEffect 11 113.00244967 32.85240820 9.905373768
3    DPDPE agonistEffect 15  44.85616454 22.20603070 5.733572472
4 DPDPE antagonistEffect 10  75.04208820 25.73318018 8.137546081
```

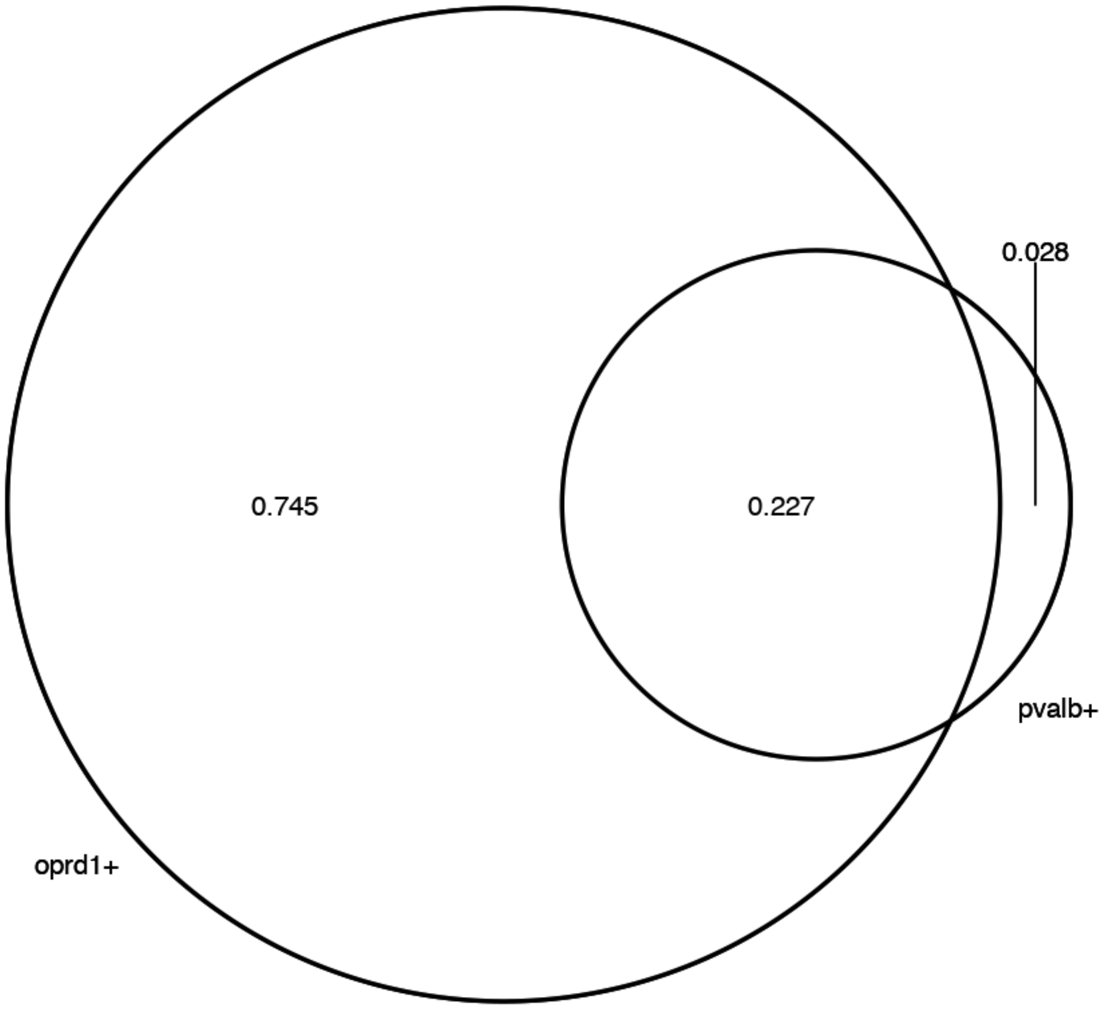
8.2 Figure 5f

In [26]:
```
##########################################################################################
#     Figure 5f
##########################################################################################
# plot venn diagram
# based on data: DOR PV ACC Counting.xls

IHC_PV_mean = 0.196
IHC_DOR_mean = 0.990
IHC_PVDOR_mean = 0.186

IHC_PV_stdev = 0.005
```

```
IHC_DOR_stdev = 0.001
IHC_PVDOR_stdev = 0.006

ISH_pvalb_mean = 0.255
ISH_oprd1_mean = 0.972
ISH_pvalboprd1_mean =         0.227

ISH_pvalb_stdev = 0.029
ISH_oprd1_stdev = 0.005
ISH_pvalboprd1_stdev =        0.034

vennIHC.plot <- draw.pairwise.venn(IHC_PV_mean, IHC_DOR_mean, IHC_PVDOR_mean, c("PV+"

grid.draw(vennIHC.plot)
```



```
In [27]: vennISH.plot <- draw.pairwise.venn(ISH_pvalb_mean, ISH_oprd1_mean, ISH_pvalboprd1_mea
         grid.draw(vennISH.plot)
```



9 Figure 6

See analyze_cellAttached_experiments.m

10 Figure 6 - figure supplement 1

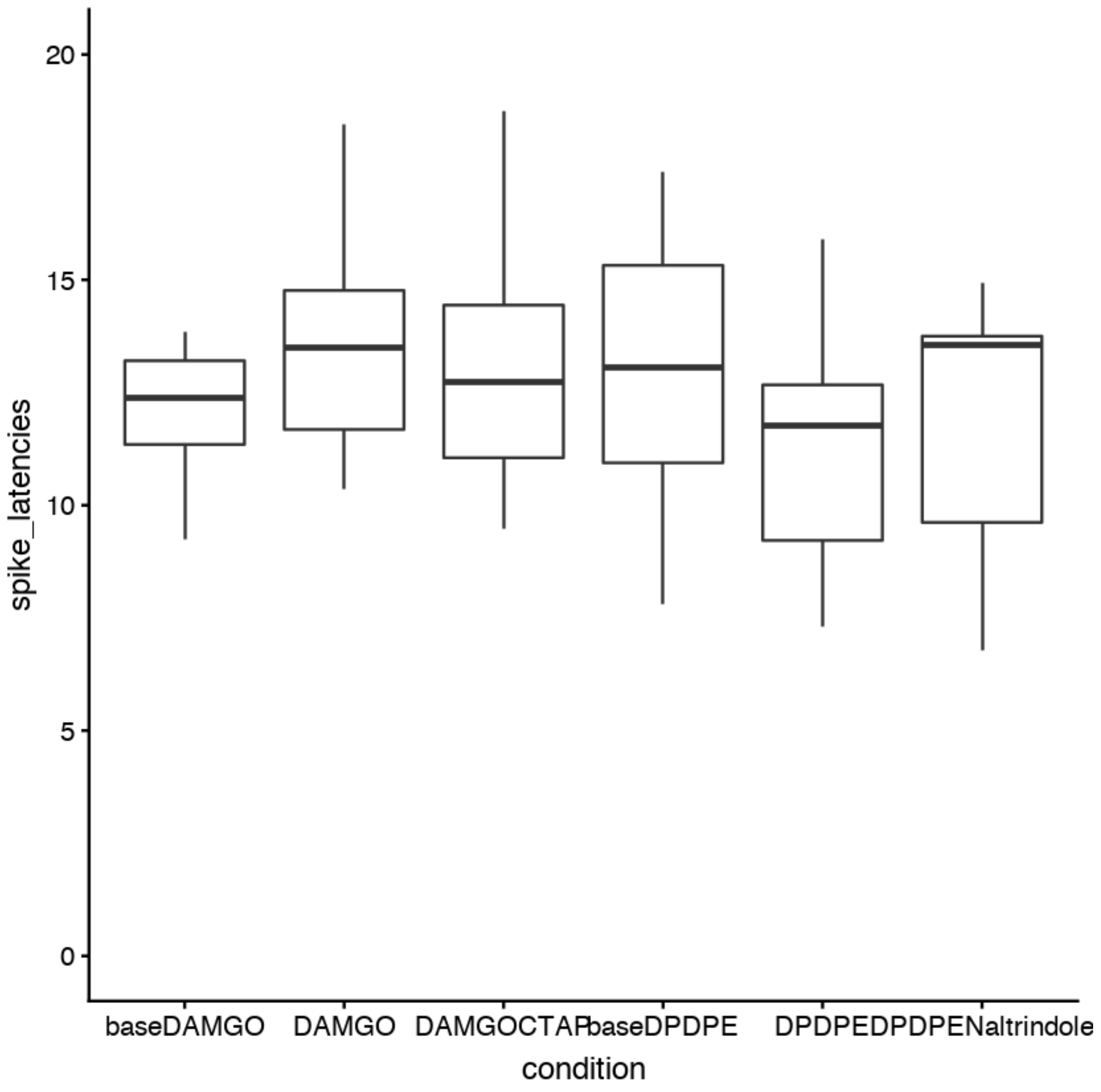
```
In [28]: ##########################################################################################
         #      Figure 6 - figure supplement 1
```

```
###########################################################################
# Plot latency data and perform Test for Significance on spike probability
# across conditions
# Requires installation of packages: 'Skillings.Mack', 'ggplot2', 'cowplot'
library(Skillings.Mack)
library(ggplot2)
library(cowplot)
df_latency = read.csv('data/data_cellAttached_latencies.csv')

# subset for all DAMGO experiments with a baseline value
df_DAMGO = df_latency[rep(!is.nan(df_latency$spike_latency_DAMGO[grep('baseDAMGO', df_
print('DAMGO AP latency Skillings-Mack analysis')
Ski.Mack(df_DAMGO$spike_latency_DAMGO,df_DAMGO$conditionDAMGO,df_DAMGO$cellDAMGO)
print('Paired Wilcoxon signed rank test baseline vs Damgo')
# define here the two groups to compare
first_group = df_DAMGO$spike_latency_DAMGO[df_DAMGO$conditionDAMGO=='baseDAMGO']
second_group = df_DAMGO$spike_latency_DAMGO[df_DAMGO$conditionDAMGO=='DAMGO']
# Below is required to calculate n and statistic in a sequence independent way
# (first vs second group and second vs first group)
wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
diff = c(first_group-second_group)
n = length(na.omit(diff[diff != 0]))
print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))

print('Paired Wilcoxon signed rank test Damgo vs CTAP')
# define here the two groups to compare
first_group = df_DAMGO$spike_latency_DAMGO[df_DAMGO$conditionDAMGO=='DAMGO']
second_group = df_DAMGO$spike_latency_DAMGO[df_DAMGO$conditionDAMGO=='DAMGOCTAP']
# Below is required to calculate n and statistic in a sequence independent way
# (first vs second group and second vs first group)
wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
diff = c(first_group-second_group)
n = length(na.omit(diff[diff != 0]))
print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))

# subset for all DPDPE experiments with a baseline value
df_DPDPE = df_latency[rep(!is.nan(df_latency$spike_latency_DPDPE[grep('baseDPDPE', df_
print('DPDPE AP latency Skillings-Mack analysis')
Ski.Mack(df_DPDPE$spike_latency_DPDPE,df_DPDPE$conditionDPDPE,df_DPDPE$cellDPDPE)
print('Paired Wilcoxon signed rank test baseline vs DPDPE')
# define here the two groups to compare
first_group = df_DPDPE$spike_latency_DPDPE[df_DPDPE$conditionDPDPE=='baseDPDPE']
second_group = df_DPDPE$spike_latency_DPDPE[df_DPDPE$conditionDPDPE=='DPDPE']
# Below is required to calculate n and statistic in a sequence independent way
```

```
        # (first vs second group and second vs first group)
        wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
        wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
        statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
        diff = c(first_group-second_group)
        n = length(na.omit(diff[diff != 0]))
        print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))
        print('Paired Wilcoxon signed rank test DPDPE vs Naltrindole')
        # define here the two groups to compare
        first_group = df_DPDPE$spike_latency_DPDPE[df_DPDPE$conditionDPDPE=='DPDPE']
        second_group = df_DPDPE$spike_latency_DPDPE[df_DPDPE$conditionDPDPE=='DPDPENaltrindole
        # Below is required to calculate n and statistic in a sequence independent way
        # (first vs second group and second vs first group)
        wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
        wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
        statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
        diff = c(first_group-second_group)
        n = length(na.omit(diff[diff != 0]))
        print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))

        #for plotting combine condition and spike_latency columns.
        spike_latencies = c(df_DAMGO$spike_latency_DAMGO,df_DPDPE$spike_latency_DPDPE)
        conditions = factor(c(as.character(df_DAMGO$conditionDAMGO),as.character(df_DPDPE$conc
        df_plot = data.frame('condition'=conditions, 'spike_latency'=spike_latencies)
        ggplot(data=df_plot, aes(x=condition, y=spike_latencies))+
            geom_boxplot()+
            coord_cartesian(ylim=c(0, 20))

[1] "DAMGO AP latency Skillings-Mack analysis"

Skillings-Mack Statistic =  2.000000 , p-value =  0.367879
Note: the p-value is based on the chi-squared distribution with d.f. =  2

$Nblocks
[1] 7

$Ntreatments
[1] 3

$rawdata
             [,1]        [,2]        [,3]        [,4]        [,5]         [,6]
[1,]  9.240740741 11.28064516 13.33684211 13.07600000 11.41578947 12.386486486
[2,] 12.900000000 10.36896552 13.50000000 15.04000000 10.46000000 14.490000000
[3,] 10.995833333 11.10975610 13.43636364 15.44285714 12.72000000  9.479487179
            [,7]
[1,] 13.86000000
[2,] 18.46666667
[3,] 18.73333333
```

```
$rankdata
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    1    3    1    1    2    2    1
[2,]    3    1    3    2    1    3    2
[3,]    2    2    2    3    3    1    3

$varCovarMatrix
     [,1] [,2] [,3]
[1,]   14   -7   -7
[2,]   -7   14   -7
[3,]   -7   -7   14

$adjustedSum
             [,1]        [,2]        [,3]
[1,] -5.196152423 1.732050808 3.464101615

[1] "Paired Wilcoxon signed rank test baseline vs Damgo"
[1] "W( 7 ) =  5 . p =  0.15625"
[1] "Paired Wilcoxon signed rank test Damgo vs CTAP"
[1] "W( 7 ) =  13 . p =  0.9375"
[1] "DPDPE AP latency Skillings-Mack analysis"

Skillings-Mack Statistic =  1.750000 , p-value =  0.416862
Note: the p-value is based on the chi-squared distribution with d.f. =  2

$Nblocks
[1] 8

$Ntreatments
[1] 3

$rawdata
            [,1]        [,2]        [,3]        [,4]        [,5]         [,6]
[1,] 8.590625000 7.810526316 11.72500000 13.40714286 17.40000000 12.685714286
[2,] 7.317910448 8.235000000 12.40952381 13.45416667 12.40370370  9.547222222
[3,] 6.786301370 9.591666667 13.81818182 13.49615385 13.64285714  9.627777778
            [,7]        [,8]
[1,] 14.80714286 16.86666667
[2,] 11.10277778 15.88750000
[3,] 13.72857143 14.93333333

$rankdata
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    3    1    1    1    3    3    3    3
[2,]    2    2    2    2    1    1    1    2
[3,]    1    3    3    3    2    2    2    1
```

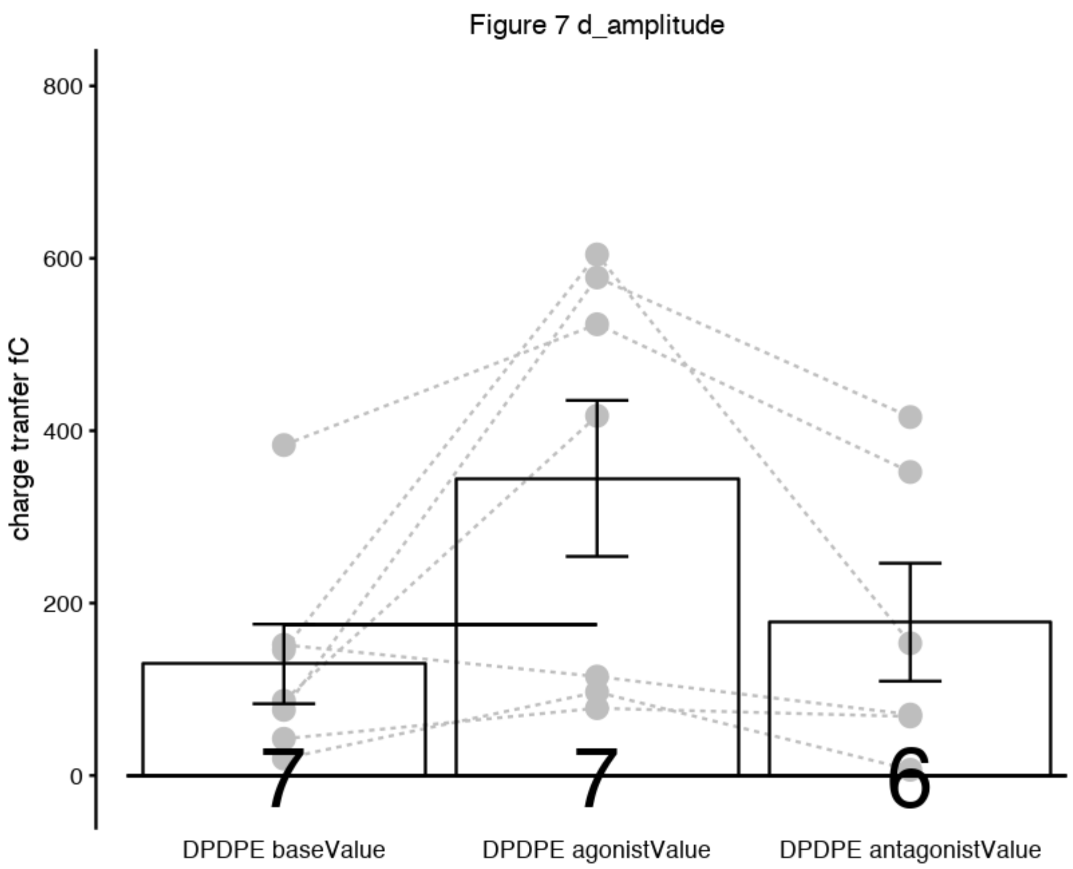
```
$varCovarMatrix
     [,1] [,2] [,3]
[1,]   16   -8   -8
[2,]   -8   16   -8
[3,]   -8   -8   16

$adjustedSum
            [,1]         [,2]        [,3]
[1,] 3.464101615 -5.196152423 1.732050808

[1] "Paired Wilcoxon signed rank test baseline vs DPDPE"
[1] "W( 8 ) =  6 . p =  0.109375"
[1] "Paired Wilcoxon signed rank test DPDPE vs Naltrindole"
[1] "W( 8 ) =  7 . p =  0.1484375"
```



11 Figure 7d

In [29]:
```
##########################################################################################
#     Figure 7 d
##########################################################################################
# Subset: optogentic stim in ACC, recorded EPSC, from MSNs with DPDPE
tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DPDPE")

#Important for good functioning of the script: No need to change:
cols = colnames(tempSDF)
tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])

# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'area' )
sDf$value[sDf$antagonistName =="DPDPEICI" & sDf$variable =="antagonistEffect" ] = NA
sDf$value[sDf$antagonistName =="DPDPENALOX" & sDf$variable =="antagonistEffect" ] = NA
xLabel = c("DPDPE","DPDPENaltri")
antagonistSelect = c("DPDPENALTRI")
graphTitle = c("Figure 7 d")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1]){
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)

# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)

SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)

reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ra
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num
sDf$groupnames = factor(paste(sDf$agonistName, sDf$variable) , levels=c('DPDPE baseVal
# plot the area by raw values
cols = colnames(sDf)
sDf = subset(sDf, variable =="agonistEffect")
sDf = melt(sDf, id.vars = cols[c(1:14,16,18,22)], measure.vars = cols[c(15,17,19)] )
sDf$value = sDf$value*-1
sDf$groupnames = factor(paste(sDf$agonistName, sDf$variable) , levels=c('DPDPE baseVal
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
  coord_cartesian(ylim = c(-20,800))+
  theme_cowplot(font_size = 12)+
  stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
  labs(ylab('charge tranfer fC'))+
  labs(xlab(NULL))+
  labs(title = paste(graphTitle,"amplitude",sep='_'))+
  labs(caption = paste(reportString2,'\n',sumRepOut))+
  theme(plot.caption = element_text(size = 8, hjust = 0))+
  theme(plot.title = element_text(size = 12))+
  geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
  geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
  theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
  theme(text=element_text(size=12))
```

```
Warning message:
Removed 1 rows containing non-finite values (stat_summary).Warning message:
Removed 1 rows containing missing values (geom_point).Warning message:
Removed 1 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  6.857051 , p-value =  0.032435
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 7 )=  2  p-value =  0.046875
 Baseline vs  DPDPENaltri  : Statistic: W( 4 )=  4  p-value =  0.875
 DPDPE  vs  DPDPENaltri  : Statistic: W( 4 )=  0  p-value =  0.125

 Number animals agonist:  6
 Number animals antagonist:  5
          groupnames n       value          sd         sem
 1       DPDPE baseValue 7 129.4003286 122.1391010 46.16424096
 2    DPDPE agonistValue 7 344.6776286 239.5973499 90.55928609
 3 DPDPE antagonistValue 6 177.8630833 167.5378501 68.39704090
```

12 Figure 7 - figure supplement 1

12.1 Figure 7- figure supplement 1a

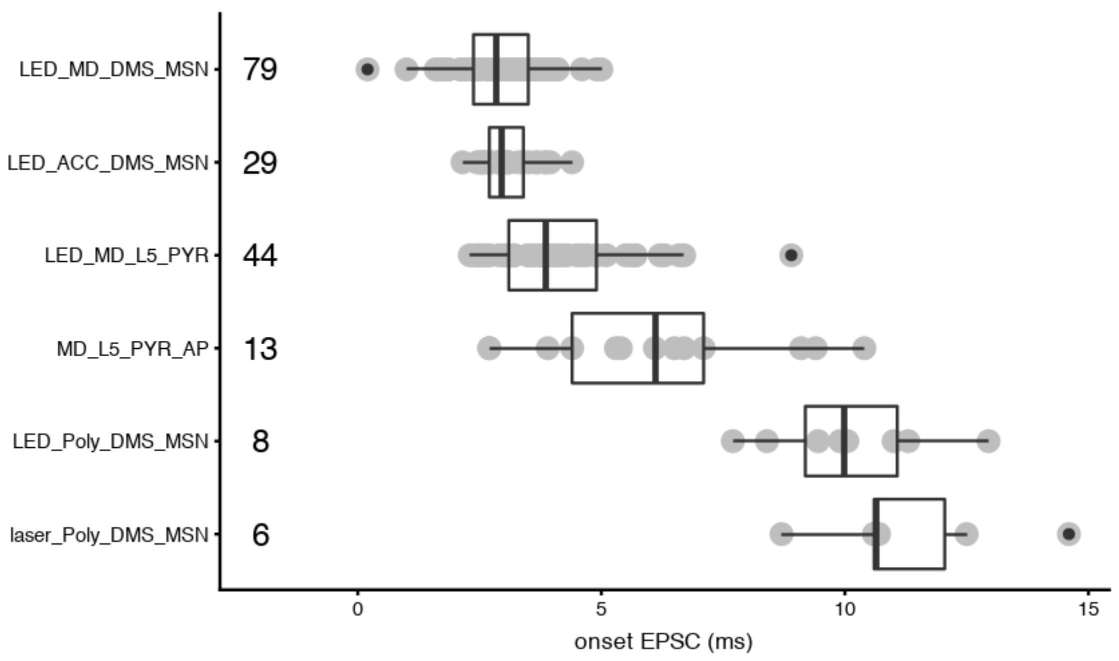
```
In [30]: ###############################################################################
         ##    Figure 7 - figure supplement 1a
         ###############################################################################
         graphTitle = 'Figure 7 - figure supplement 1a'
         # Subset: all subgroups of synaptic contact within the thalamo-cortico-striatal circu
         tempSDF = subset(cdf, genoType != "MORKOho" & genoType != "MORWTho"  & stimSource == '
         tempSDF1 = subset(tempSDF, cellID == 'will20170405c001' & conditionName == 'BIC');
         tempSDF2 = subset(tempSDF, cellID == 'will20170309c007' & conditionName == 'BASEDPDPE
         tempSDF3 = subset(tempSDF, cellID == 'will20170922c002' & conditionName == 'BASE')
```

```
tempSDF4 = subset(tempSDF, cellID == 'will20171107c006' & conditionName == 'SR')
tempSDF5 = subset(tempSDF, cellID == 'will20170308c000' & conditionName == 'DPDPE')
tempSDF6 = subset(tempSDF, cellID == 'will20170620c003' & conditionName == 'DPDPE')
tempSDF7 = subset(tempSDF, cellID == 'will20170620c007' & conditionName == 'DPDPENTD'
tempSDF8 = subset(tempSDF, stimChannel == 'laser')
tempSDF = rbind(tempSDF1,tempSDF2,tempSDF3,tempSDF4,tempSDF5,tempSDF6,tempSDF7,tempSD
tempSDF$channelCircuit=paste(tempSDF$stimChannel,tempSDF$circuit,sep='_')
tempSDF$channelCircuit= gsub("Poly_Poly_DMS_MSN","LED_Poly_DMS_MSN",tempSDF$channelCi

tempSDFMDACC = subset(cdf, genoType != "MORKOho" & genoType != "MORWTho"  & stimSourc
tempSDFMDACC = tempSDFMDACC[grep("BASE",tempSDFMDACC$conditionName, perl=TRUE, ignore
tempSDFMDACC$channelCircuit=paste(tempSDFMDACC$stimSource,tempSDFMDACC$circuit,sep='_
tempSDFMDACC$channelCircuit= gsub("MD_MD_L5_PYR","LED_MD_L5_PYR",tempSDFMDACC$channel

tempSDFACCDMS = subset(cdf, genoType != "MORKOho" & genoType != "MORWTho"  & stimSour
tempSDFACCDMS = tempSDFACCDMS[grep("BASE",tempSDFACCDMS$conditionName, perl=TRUE, ign
tempSDFACCDMS$channelCircuit=paste(tempSDFACCDMS$stimSource,tempSDFACCDMS$circuit,sep
tempSDFACCDMS$channelCircuit= gsub("ACC_ACC_DMS_MSN","LED_ACC_DMS_MSN",tempSDFACCDMS$

tempSDFMDDMS = subset(cdf, genoType != "MORKOho" & genoType != "MORWTho"  & stimSourc
tempSDFMDDMS = tempSDFMDDMS[grep("BASE",tempSDFMDDMS$conditionName, perl=TRUE, ignore
tempSDFMDDMS$channelCircuit=paste(tempSDFMDDMS$stimSource,tempSDFMDDMS$circuit,sep='_
tempSDFMDDMS$channelCircuit= gsub("MD_MD_DMS_MSN","LED_MD_DMS_MSN",tempSDFMDDMS$chann

tempSDF = rbind(tempSDF,tempSDFMDACC,tempSDFMDDMS,tempSDFACCDMS)
tempSDF$synapse = tempSDF$stimSource;
tempSDF$synapse = gsub("MD","mono",tempSDF$synapse)
tempSDF$synapse = gsub("ACC","mono",tempSDF$synapse)
tempSDF$charge0Amplitude = tempSDF$area/tempSDF$amplitude
tempSDF$test1DrugName = NULL
tempSDF =rbind(tempSDF,apdf)

synapseWilcox = wilcox.test(tempSDF$onset[grep("MONO",tempSDF$synapse,perl=TRUE, igno
synapseWilcoxFlip = wilcox.test(tempSDF$onset[grep("POLY",tempSDF$synapse,perl=TRUE, 
synapseWilcox$statistic = min(c(synapseWilcox$statistic,synapseWilcoxFlip$statistic))
synapseAllDunn = capture.output(dunn.test(tempSDF$onset,tempSDF$channelCircuit, kw = 


reportString = paste(synapseAllDunn)
reportString3 = ''
for (ii in 1:length(levels(data_summary$channelCircuit))){
  animalDf = subset(tempSDF, channelCircuit == data_summary$channelCircuit[ii])
  animalN = length(unique(animalDf$animalID))
  reportString3 = paste(reportString3,data_summary$channelCircuit[ii],' :',animalN,' 
}
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.fini
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.f
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Nu
data_summary = data_meanSDSEM(tempSDF, varname = "onset", groupnames = "channelCircui
data_summary$channelCircuit = factor(data_summary$channelCircuit, levels=c("laser_Pol
tempSDF$channelCircuit = factor(tempSDF$channelCircuit, levels=c("laser_Poly_DMS_MSN"
factor(data_summary)
levels(data_summary$channelCircuit)
sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}
reportString = '';
for (ii in 1:length(synapseAllDunn)){
  reportString = paste(reportString,synapseAllDunn[ii],'\n', sep=' ')
}
reportString2 = paste('Mann-whitney U Tests Mono vs Poly synaptic: Statistic: W= ',sy

ggplot(tempSDF, aes(x = channelCircuit , y = onset))+
  geom_point(data = tempSDF, aes(x = channelCircuit , y = onset), fill = "gray", colo
  geom_boxplot(varwidth = FALSE, notch = FALSE, fill = NA, )+
  theme_cowplot(font_size =10)+
  stat_summary(data =tempSDF, fun.data = give.n, geom = "text",fun.y = median, positi
  labs(ylab('onset EPSC (ms)'))+
  labs(xlab(NULL))+
  labs(caption = paste(reportString,reportString2,'\n', reportString3,'\n',sumRepOut)
  theme(plot.caption = element_text(size = 5, hjust = 0))+
  theme(text=element_text(size=10))+
  coord_flip()
```

channelCircuit c(1, 5, 6, 4, 2, 3) **n** c(6, 29, 79, 44, 8, 13) **onset** c(11.2833333333333, 3.07758620689655, 2.90379746835443, 4.13409090909091, 10.09375, 6.13076923076923) **sd** c(2.02130320997783, 0.525354677285558, 0.851724199668649, 1.41847333291624, 1.66849750631262, 2.44723810835283) **sem** c(0.825193579982567, 0.0975559144614377, 0.0958264591973098, 0.213842900927638, 0.589902950553249, 0.678741729456588)

Levels: 1. 'c(1, 5, 6, 4, 2, 3)' 2. 'c(6, 29, 79, 44, 8, 13)' 3. 'c(11.2833333333333, 3.07758620689655, 2.90379746835443, 4.13409090909091, 10.09375, 6.13076923076923)' 4. 'c(2.02130320997783, 0.525354677285558, 0.851724199668649, 1.41847333291624, 1.66849750631262, 2.44723810835283)' 5. 'c(0.825193579982567, 0.0975559144614377, 0.0958264591973098, 0.213842900927638, 0.589902950553249, 0.678741729456588)'

1. 'laser_Poly_DMS_MSN' 2. 'LED_Poly_DMS_MSN' 3. 'MD_L5_PYR_AP' 4. 'LED_MD_L5_PYR' 5. 'LED_ACC_DMS_MSN' 6. 'LED_MD_DMS_MSN'



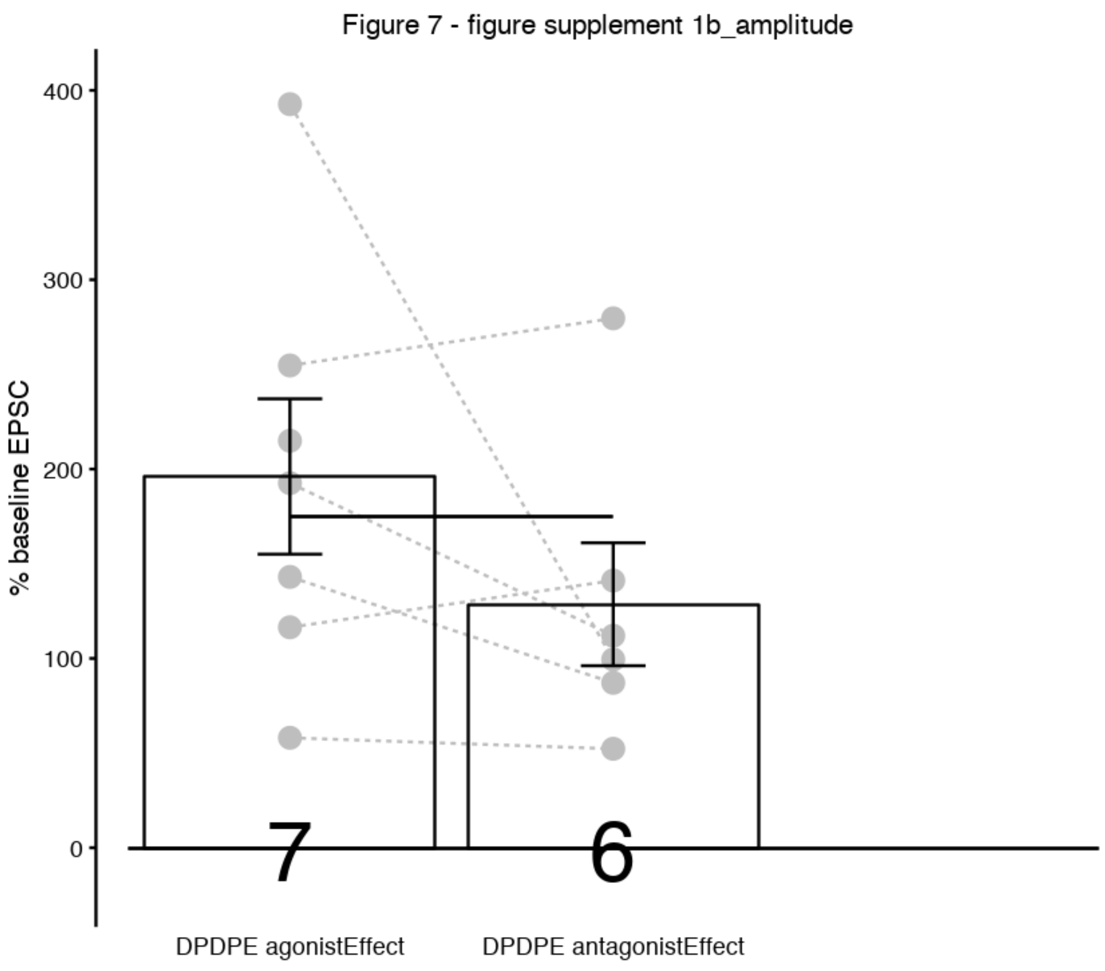
```
  Kruskal-Wallis rank sum test

data: x and group
Kruskal-Wallis chi-squared = 73.2833, df = 5, p-value = 0


                Comparison of x by group
                      (Bonferroni)
Col Mean-|
Row Mean |   laser_Po   LED_ACC_   LED_MD_D   LED_MD_L   LED_Poly
---------+-------------------------------------------------------
LED_ACC_ |   4.359364
         |    0.0001*
         |
LED_MD_D |   4.953977   0.657185
         |    0.0000*     1.0000
         |
LED_MD_L |   2.891991  -2.912307  -4.461592
         |     0.0287     0.0269    0.0001*
         |
LED_Poly |   0.108716  -4.748801  -5.495978  -3.121775
         |     1.0000    0.0000*    0.0000*    0.0135*
         |
MD_L5_PY |   1.416015  -3.763880  -4.674144  -1.773048   1.424604
         |     1.0000    0.0013*    0.0000*     0.5717     1.0000

alpha = 0.05
Reject Ho if p <= alpha/2
Mann-whitney U Tests Mono vs Poly synaptic: Statistic: W=  15  p-value =  9.18461493019699e-10
  : 0  animals   : 0  animals
    channelCircuit  n      onset          sd         sem
1 laser_Poly_DMS_MSN  6 11.283333333 2.0213032100 0.82519357998
2   LED_ACC_DMS_MSN 29  3.077586207 0.5253546773 0.09755591446
3    LED_MD_DMS_MSN 79  2.903797468 0.8517241997 0.09582645920
4     LED_MD_L5_PYR 44  4.134090909 1.4184733329 0.21384290093
5  LED_Poly_DMS_MSN  8 10.093750000 1.6684975063 0.58990295055
6      MD_L5_PYR_AP 13  6.130769231 2.4472381084 0.67874172946
```

12.2 Figure 7 - figure supplement 1b

```
In [31]: ##########################################################################
         #     Figure 7 - figure supplement 1b
         ##########################################################################
         # Subset: opto stim in ACC, recorded EPSC, from MSNs with DPDPE
         tempSDF = subset(df, genoType != "MORKOho" & genoType != "MORWTho" & genoType != "vGlu
         tempSDF = subset(tempSDF, signal == "EPSC" & agonistName == "DPDPE")

         #Important for good functioning of the script: No need to change:
         cols = colnames(tempSDF)
         tempSDF = melt(tempSDF, id.vars = cols[1:19],measure.vars = cols[20:21])
```

```
# give the tempSDF a specific name, needs to be changed for every new subset
sDf = subset(tempSDF, parameter == 'amplitude' )
sDf$value[sDf$antagonistName =="DPDPEICI" & sDf$variable =="antagonistEffect" ] = NA
sDf$value[sDf$antagonistName =="DPDPENALOX" & sDf$variable =="antagonistEffect" ] = N
xLabel = c("DPDPE","DPDPENaltri")
antagonistSelect = c("DPDPENALTRI")
graphTitle = c("Figure 7 - figure supplement 1b")

# prepare data into matrix for Skillings-Mack test.
ssDf = subset(sDf, variable == "agonistEffect" | variable == "antagonistEffect")
ssDf = subset(ssDf, is.finite(baseValue)& is.finite(agonistValue))
# # to avoid problems with sign reversal during agonist/antagonist treatment all cond
# # after that the SMmatrix can be transformed to absolute values
for (u in 1:nrow(ssDf)){
  if (ssDf$baseValue[u]<0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]>0){
        ssDf$agonistValue[u]=-0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]>0){
        ssDf$antagonistValue[u]=-0.0001
      }
    }
  }
  if (ssDf$baseValue[u]>0){
    if (is.finite(ssDf$agonistValue[u])){
      if (ssDf$agonistValue[u]<0){
        ssDf$agonistValue[u]=0.0001
      }
    }
    if (is.finite(ssDf$antagonistValue[u])){
      if (ssDf$antagonistValue[u]<0){
        ssDf$antagonistValue[u]=0.0001
      }
    }
  }
  if (is.finite(ssDf$antagonistName[u])){
    if (ssDf$antagonistName[u] !=antagonistSelect[1]){
      ssDf$antagonistValue[u] = NA
    }
  }
}
ssDf$baseValue = abs(ssDf$baseValue)
ssDf$agonistValue =abs(ssDf$agonistValue)
ssDf$antagonistValue = abs(ssDf$antagonistValue)
```

```
# DPDPE
##########
SMlen = length(unique(ssDf$cellID[ssDf$agonistName == 'DPDPE']))
SMmatrix = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$agonistValue[s
Gs = c(rep('baseline',SMlen),rep('agonist',SMlen),rep('antagonist',SMlen))
Bs = rep(1:SMlen,3)
SMresult = capture.output(Ski.Mack(SMmatrix,groups=Gs, blocks=Bs))
BAgWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$agon
BAgWilcoxFlip = wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ss
BAgWilcox$statistic = min(c(BAgWilcox$statistic, BAgWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$agonistValue[ssDf
BAgWilcox$n = length(na.omit(diff[diff !=0]))

BAnWilcox = wilcox.test(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen],ssDf$anta
BAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen]
BAnWilcox$statistic = min(c(BAnWilcox$statistic, BAnWilcoxFlip$statistic))
diff = c(ssDf$baseValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValue[s
BAnWilcox$n = length(na.omit(diff[diff !=0]))

AgAnWilcox= wilcox.test(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen], ssDf$
AgAnWilcoxFlip = wilcox.test(ssDf$antagonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen
AgAnWilcox$statistic = min(c(AgAnWilcox$statistic, AgAnWilcoxFlip$statistic))
diff = c(ssDf$agonistValue[ssDf$agonistName == 'DPDPE'][1:SMlen] - ssDf$antagonistValu
AgAnWilcox$n = length(na.omit(diff[diff !=0]))

BAgAnno = pvalAnno(BAgWilcox$p.value,1)
BAnAnno = pvalAnno(BAnWilcox$p.value,1)
AgAnAnno =  pvalAnno(AgAnWilcox$p.value,1)
reportString = paste(SMresult[2],'\n','Paired Wilcoxon-rank Tests (Wilcoxon signed-ran
animalDf = subset(sDf, variable == 'agonistEffect' & agonistName == 'DPDPE' & is.finit
agonistAnimal = length(unique(animalDf$animalID))
animalDf = subset(sDf, variable == 'antagonistEffect' & agonistName == 'DPDPE' & is.fi
antagonistAnimal = length(unique(animalDf$animalID))
reportString2 = paste(reportString,'\n Number animals agonist: ',agonistAnimal,'\n Num
sDf$groupnames = paste(sDf$agonistName, sDf$variable)
data_summary = data_meanSDSEM(sDf, varname = "value", groupnames = "groupnames")
sumOut = capture.output(data_summary)
sumRepOut = sumOut[1]
for(u in 2:length(sumOut)){
  sumRepOut = paste(sumRepOut,'\n', sumOut[u])
}

ggplot(data_summary, aes(x = groupnames , y = value))+
  geom_point(data = sDf, aes(x = groupnames , y = value), fill = "gray", colour = "gra
  geom_line(data = sDf, aes(group = cellID ), lty = 2, colour = "gray")+
  geom_bar(stat = "identity", position = position_dodge(), colour = "black", fill = NA
  geom_errorbar(aes(ymin=value-sem, ymax=value+sem), width=.2, position=position_dodge
  coord_cartesian(ylim = c(-20,400))+
           theme_cowplot(font_size = 12)+
           stat_summary(data =sDf, fun.data = give.n, geom = "text",fun.y = median, position =
           labs(ylab('% baseline EPSC'))+
           labs(xlab(NULL))+
           labs(title = paste(graphTitle,"amplitude",sep='_'))+
           labs(caption = paste(reportString2,'\n',sumRepOut))+
           theme(plot.caption = element_text(size = 5, hjust = 0))+
           theme(plot.title = element_text(size = 12))+
           geom_segment(aes(x=1, y=175, xend = 2, yend = 175))+
           geom_segment(aes(x=0.5, y=0, xend = 3.5, yend = 0))+
           theme(axis.line.x = element_line(linetype='blank'), axis.ticks.x = element_blank())
           theme(text=element_text(size=12))

Warning message:
Removed 1 rows containing non-finite values (stat_summary).Warning message:
Removed 1 rows containing missing values (geom_point).Warning message:
Removed 1 rows containing missing values (geom_path).
```



```
Skillings-Mack Statistic =  3.857051 , p-value =  0.145362
 Paired Wilcoxon-rank Tests (Wilcoxon signed-rank test)
 Baseline vs  DPDPE : Statistic: W( 7 )=  3  p-value =  0.078125
 Baseline vs  DPDPENaltri  : Statistic: W( 4 )=  5  p-value =  1
 DPDPE  vs  DPDPENaltri  : Statistic: W( 4 )=  2  p-value =  0.375

 Number animals agonist:  6
 Number animals antagonist:  5
          groupnames n      value         sd        sem
 1    DPDPE agonistEffect 7 196.0710633 108.52809858 41.01976559
 2 DPDPE antagonistEffect 6 128.5885243  79.53688978 32.47079928
```

13 Rebuttal Figure 1c

```
In [32]: ##############################################################################
         #     Rebuttal figure 1c
         ##############################################################################
         # analysis of data and perform Test for Significance on EPSC
         # amplitude across conditions
         # Requires installation of packages: 'Skillings.Mack'
         library(Skillings.Mack)
         library(ggplot2)
         df = read.csv('data/data_rebuttal_fig1.csv')
         # subset for all DPDPE experiments with a baseline value
```

```
        df_DPDPE = df[rep(!is.nan(df$effect_amplitude[grep('baseNaltrindole', df$conditionDPDP
        print('Naltrindole-DPDPE effect on baseline EPSC Skillings-Mack analysis')
        Ski.Mack(df_DPDPE$effect_amplitude,df_DPDPE$conditionDPDPE,df_DPDPE$cellDPDPE)
        print('Paired Wilcoxon signed rank test baseline vs Naltrindole')
        # define here the two groups to compare
        first_group = df_DPDPE$effect_amplitude[df_DPDPE$conditionDPDPE=='baseNaltrindole']
        second_group = df_DPDPE$effect_amplitude[df_DPDPE$conditionDPDPE=='Naltrindole']
        # Below is required to calculate n and statistic in a sequence independent way
        # (first vs second group and second vs first group)
        wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
        wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
        statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
        diff = c(first_group-second_group)
        n = length(na.omit(diff[diff != 0]))
        print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))
        print('Paired Wilcoxon signed rank test Naltrindole vs DPDPENaltrindole')
        # define here the two groups to compare
        first_group = df_DPDPE$effect_amplitude[df_DPDPE$conditionDPDPE=='Naltrindole']
        second_group = df_DPDPE$effect_amplitude[df_DPDPE$conditionDPDPE=='DPDPENaltrindole']
        # Below is required to calculate n and statistic in a sequence independent way
        # (first vs second group and second vs first group)
        wilcox_antero = wilcox.test(first_group, second_group, paired = TRUE)
        wilcox_retro = wilcox.test(second_group, first_group, paired = TRUE)
        statistic = min(c(wilcox_antero$statistic, wilcox_retro$statistic))
        diff = c(first_group-second_group)
        n = length(na.omit(diff[diff != 0]))
        print(paste('W(', n, ') = ', statistic, '. p = ', wilcox_antero$p.value))
```

```
[1] "Naltrindole-DPDPE effect on baseline EPSC Skillings-Mack analysis"

Skillings-Mack Statistic =  3.714286 , p-value =  0.156118
Note: the p-value is based on the chi-squared distribution with d.f. =  2

$Nblocks
[1] 7

$Ntreatments
[1] 3

$rawdata
            [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
[1,] 1288.147967 570.3919281 550.9480322 455.4172314 690.8605557 569.1642694
[2,] 1232.114758 490.2516592 382.6019742 686.4860459 621.2439124 450.3674759
[3,] 1330.044652 453.3139970 209.8444380 565.3576579 675.4080718 313.0839955
             [,7]
[1,] 1193.1270871
[2,]  785.0636979
[3,]  752.8596021
```

```
$rankdata
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    2    3    3    1    3    3    3
[2,]    1    2    2    3    1    2    2
[3,]    3    1    1    2    2    1    1

$varCovarMatrix
     [,1] [,2] [,3]
[1,]   14   -7   -7
[2,]   -7   14   -7
[3,]   -7   -7   14

$adjustedSum
           [,1]         [,2]         [,3]
[1,] 6.92820323 -1.732050808 -5.196152423

[1] "Paired Wilcoxon signed rank test baseline vs Naltrindole"
[1] "W( 7 ) =  6 . p =  0.21875"
[1] "Paired Wilcoxon signed rank test Naltrindole vs DPDPENaltrindole"
[1] "W( 7 ) =  7 . p =  0.296875"
```

14 Rebuttal figure 2

```
In [33]: # Rebuttal figure 2 Analysis of data and perform plotting for DAMGO effect on
         # thalamostriatal projections, to check whether there is a potential
         # heterogeneity in the data that would suggest that inputs to D1 and D2 MSNs in
         # the striatum may be modulated differently.
         # import packages
         library(ggplot2)
         library(reshape2)
         library(cowplot)
         require(cowplot)
         library(tidyverse)  # data manipulation
         library(cluster)    # clustering algorithms
         library(factoextra) # clustering visualization
         library(dendextend) # for comparing two dendrograms
         library(heatmap.plus) # clustering visualization
         library(colorspace) # colors
         library(RColorBrewer)
         library(colorRamps)
         library(gplots)
         library(plyr)
         library(Skillings.Mack)

         df=read.csv('data/effectDataset.csv')
```

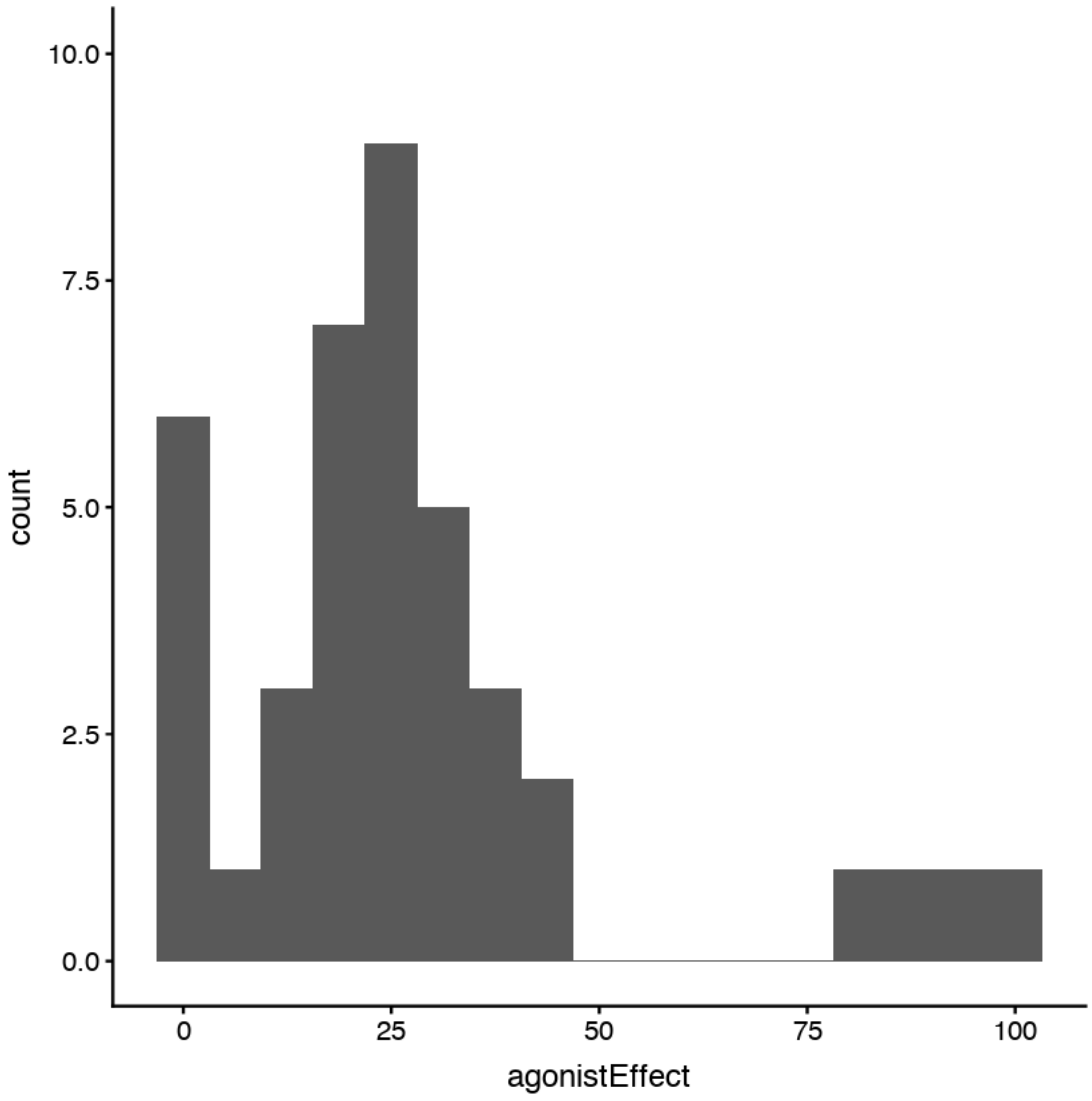
```
        df$X = NULL
        msn_df=subset(df, cellType=='MSN' & drugGroup==2)
        setHook(packageEvent("grDevices", "onLoad"),
        function(...) grDevices::X11.options(type='cairo'))
        options(device='x11')
        # subset data for thalamostriatal projections only (including MD and AM)
        thal_df = subset(msn_df, (stimSource=='MD' | stimSource=='AM') & cellID != 'will20180
```

```
 Attaching packages  tidyverse 1.2.1
 tibble  1.4.2      purrr   0.2.5
 tidyr   0.8.2      dplyr   0.7.8
 readr   1.1.1      stringr 1.3.1
 tibble  1.4.2      forcats 0.3.0
 Conflicts  tidyverse_conflicts()
 dplyr::arrange()    masks plyr::arrange()
 readr::col_factor() masks scales::col_factor()
 purrr::compact()    masks plyr::compact()
 dplyr::count()      masks plyr::count()
 purrr::discard()    masks scales::discard()
 tidyr::expand()     masks Matrix::expand()
 dplyr::failwith()   masks plyr::failwith()
 dplyr::filter()     masks stats::filter()
 cowplot::ggsave()   masks ggplot2::ggsave()
 dplyr::id()         masks plyr::id()
 dplyr::lag()        masks stats::lag()
 dplyr::mutate()     masks plyr::mutate()
 dplyr::rename()     masks plyr::rename()
 dplyr::select()     masks MASS::select()
 dplyr::summarise()  masks plyr::summarise()
 dplyr::summarize()  masks plyr::summarize()
Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ

---------------------
Welcome to dendextend version 1.9.0
Type citation('dendextend') for how to cite the package.

Type browseVignettes(package = 'dendextend') for the package vignette.
The github page is: https://github.com/talgalili/dendextend/

Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issue
Or contact: <tal.galili@gmail.com>

        To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
---------------------


Attaching package: dendextend
```

```
The following object is masked from package:VennDiagram:

    rotate

The following object is masked from package:stats:

    cutree


Attaching package: gplots

The following object is masked from package:stats:

    lowess
```

14.1 Rebuttal figure 2a

```
In [34]: ##############################################################################
         #     Rebuttal figure 2a
         ##############################################################################
         # plot distribution of DAMGO effect on thalamostriatal
         # projections.

         ggplot(data=subset(thal_df, parameter=='amplitude'), aes(agonistEffect)) +
             geom_histogram(binwidth=25/4)+
             ylim(0,10)

Warning message:
Removed 1 rows containing non-finite values (stat_bin).
```



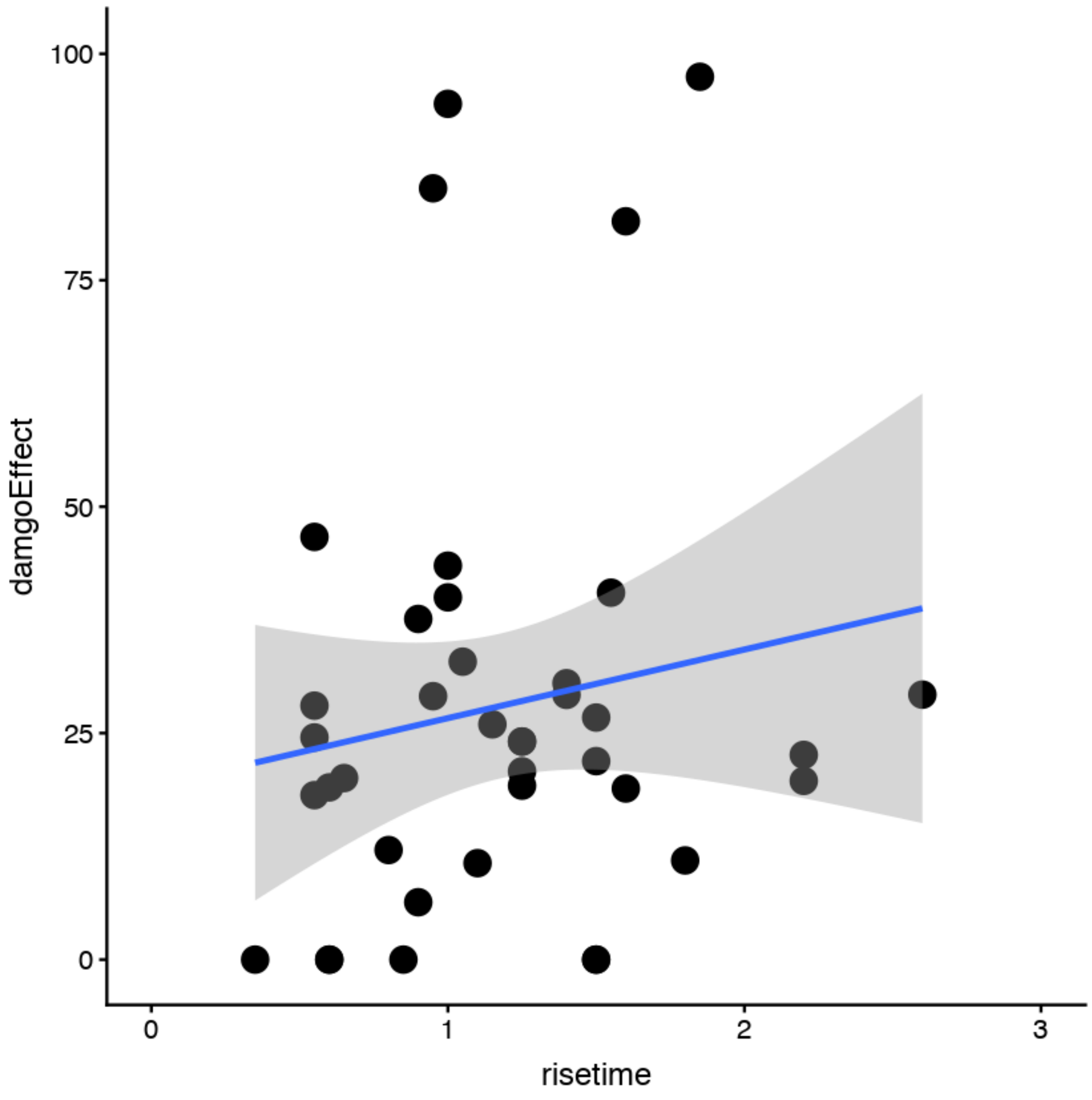
```
In [35]: print('damgoEffect normality test')
         shapiro.test(subset(thal_df, parameter=='amplitude')$agonistEffect)

[1] "damgoEffect normality test"



Shapiro-Wilk normality test

data:  subset(thal_df, parameter == "amplitude")$agonistEffect
W = 0.82149112, p-value = 1.962699e-05
```

14.2 Rebuttal figure 2b

```
In [36]: ################################################################################
         #     Rebuttal figure 2b
         ################################################################################
         # First re-organize the df to allow for subsetting
         sthal_df = thal_df[,c(1:12, 20, 15)]
         tthal_df <- subset(sthal_df, parameter=='amplitude')
         colnames(tthal_df)[colnames(tthal_df)=='baseValue'] <- 'amplitude'
         colnames(tthal_df)[colnames(tthal_df)=='agonistEffect'] <- 'damgoEffect'
         tthal_df$parameter <- NULL
         tthal_df$onset <- subset(sthal_df, parameter=='onset')$baseValue
         tthal_df$risetime <- subset(sthal_df, parameter=='risetime')$baseValue
         tthal_df$slope <- subset(sthal_df, parameter=='slope')$baseValue
         tthal_df$onset2peakTime <- subset(sthal_df, parameter=='onset2peakTime')$baseValue
         tthal_df$halfwidth <- subset(sthal_df, parameter=='halfwidth')$baseValue
         tthal_df$decay <- subset(sthal_df, parameter=='decay')$baseValue
         tthal_df$area <- subset(sthal_df, parameter=='area')$baseValue
         tthal_df$charge0Amplitude <- subset(sthal_df, parameter=='charge0Amplitude')$baseValu
         # fill out NA for onset of will20180705_732c002 with median of onset
         tthal_df[tthal_df$cellID == 'will20180705_732c002',]$onset = median(tthal_df$onset, na
         rownames(tthal_df) = 1:nrow(tthal_df)
         tthal_df_ref = tthal_df
         tthal_df = tthal_df[,-12]

In [37]: # Rebuttal figure 2b, plot distribution of baseline EPSC risetime vs DAMGO
         # effect for thalamostriatal inputs. Perform linear regression to check for
         # correlation between effect of DAMGO and risetime of the baseline EPSC.
         ggplot(data=tthal_df_ref, aes(risetime, damgoEffect))+
             geom_point(size=5)+
             geom_smooth(method=lm, se=TRUE)+
             ylim(0,100)+
             xlim(0,3)
         model = lm(risetime ~ damgoEffect, data=tthal_df_ref)
         summary(model)
```

```
Warning message:
Removed 2 rows containing non-finite values (stat_smooth).Warning message:
Removed 2 rows containing missing values (geom_point).Warning message in grid.Call.graphics(C_
semi-transparency is not supported on this device: reported only once per page


Call:
lm(formula = risetime ~ damgoEffect, data = tthal_df_ref)

Residuals:
        Min          1Q      Median          3Q         Max
-0.78276215 -0.44111118 -0.06654981  0.28960441  1.83244029
```

```
Coefficients:
               Estimate  Std. Error t value   Pr(>|t|)
(Intercept) 1.132762132 0.143774537 7.87874 1.6452e-09 ***
damgoEffect 0.003210778 0.003902538 0.82274    0.41579
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.5924341 on 38 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.01750148,Adjusted R-squared:  -0.008353749
F-statistic: 0.6769029 on 1 and 38 DF,  p-value: 0.4157885
```



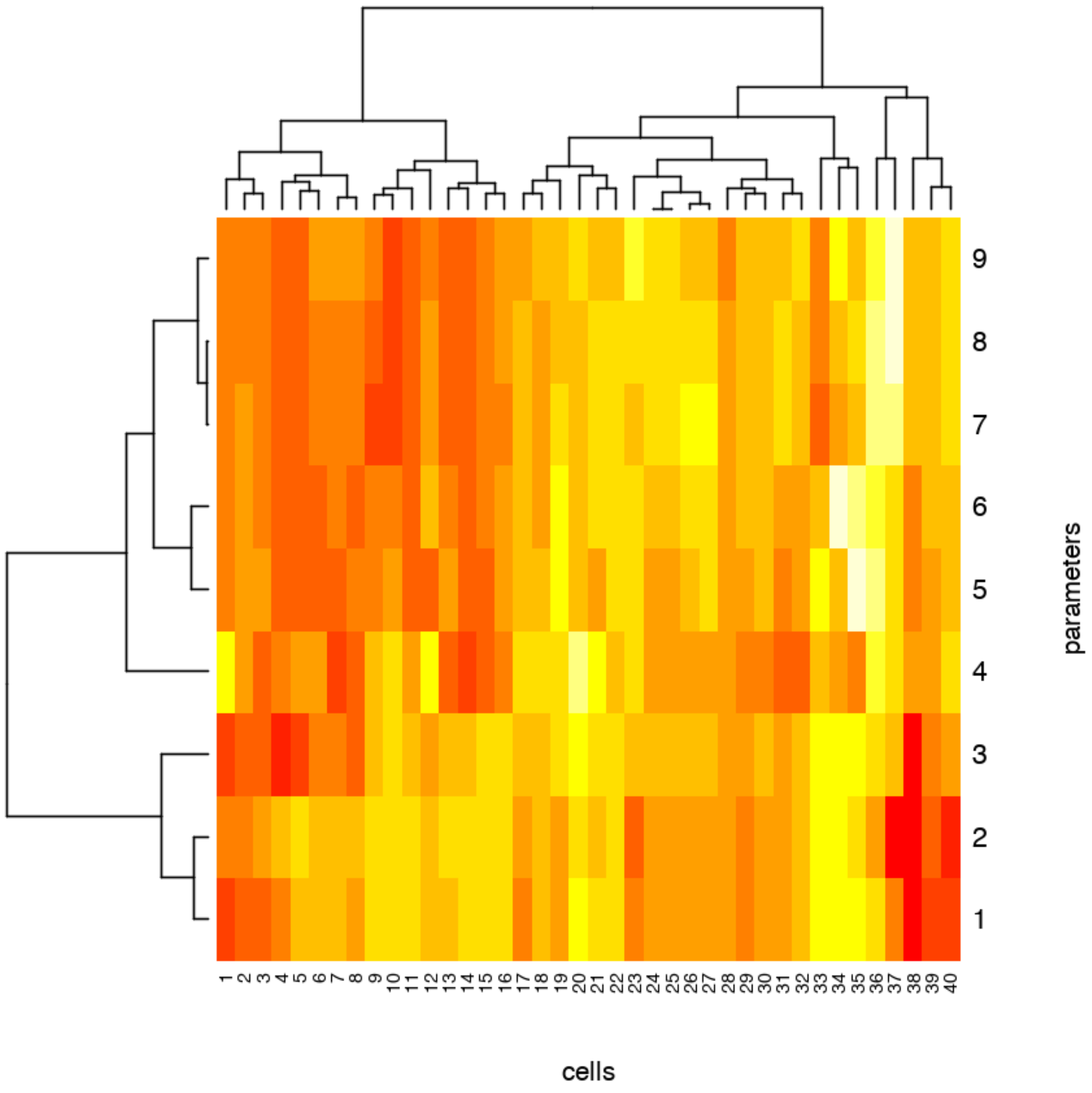
14.3 Rebuttal figure 2c-d

```
In [38]: ##################################################################################
         #     Rebuttal figure 2c and d
         ##################################################################################
         # Define two subpopulations based on the baseline EPSC
         # parameters. Based on those two subpopulations, test whether DAMGO effect on
         # baseline is significantly different between the two populations.
         # Generate a matrix with all parameters and make sure that no constant or zero
         # columns are present
         input_thal = tthal_df[12:ncol(tthal_df)]
         input_thal = na.omit(input_thal)
         which(apply(input_thal, 2, var)==0) # some variables have a constant or zero column,
         input_thal = input_thal[, apply(input_thal, 2, var)!=0]
         input_thal = scale(input_thal, center=TRUE, scale=TRUE) # center to the mean value an
         # generate distance matrixes for the cells and parameters:
         d_HCA_thal_param = get_dist(t(input_thal), method="pearson")
         d_HCA_thal_cell = get_dist((input_thal), method="euclidean")
         # Check that for both matrixes which clustering methods produces the strongest
         # clusters
         m <- c( "average", "single", "complete", "ward")
         names(m) <- c( "average", "single", "complete", "ward")
         ac <- function(x) {
           agnes(d_HCA_thal_cell, method = x)$ac
         }
         map_dbl(m, ac)
         m <- c( "average", "single", "complete", "ward")
         names(m) <- c( "average", "single", "complete", "ward")
         ac <- function(x) {
           agnes(d_HCA_thal_param, method = x)$ac
         }
         map_dbl(m, ac)
```

average 0.738408243014765 **single** 0.597530680588402 **complete** 0.815356990926301 **ward** 0.887360896778015
average 0.802350726435561 **single** 0.718641887706935 **complete** 0.860649035675753 **ward** 0.886143224910785

```
In [39]: # Based on the comparison of methods it turns out that Ward provides the strongest
         # clusters for both cell and parameter clustering. So continue with Ward.
         HCA_thal_cell = agnes(d_HCA_thal_cell, method = "ward")
         HCA_thal_param = agnes(d_HCA_thal_param, method = "ward")
         heatmap(t(input_thal), Colv=as.dendrogram(HCA_thal_cell), Rowv=as.dendrogram(HCA_thal
         HCA_thal_cell$order[1:15]
         HCA_thal_param$order.lab
```

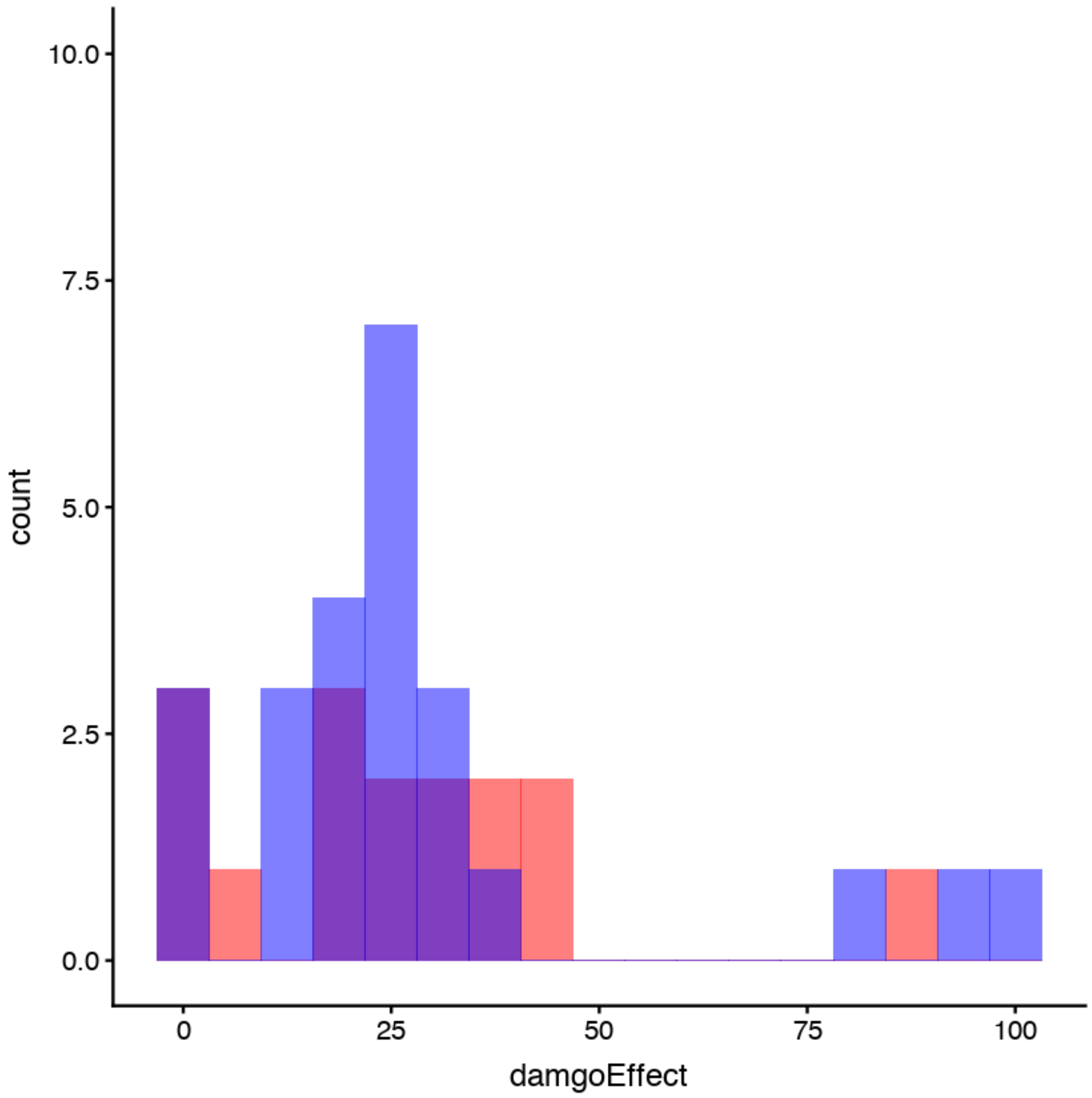
1. 1 2. 18 3. 36 4. 16 5. 20 6. 33 7. 29 8. 30 9. 4 10. 5 11. 17 12. 8 13. 19 14. 27 15. 28
1. 'amplitude' 2. 'area' 3. 'slope' 4. 'onset' 5. 'risetime' 6. 'onset2peakTime' 7. 'halfwidth' 8. 'chargeOAmplitude' 9. 'decay'



```
In [40]: # Plot and test whether the two putative subpopulations have a difference in DAMGO ef
         tthal_df_ref$cluster = rep(1,nrow(tthal_df_ref))
         clust1 = HCA_thal_cell$order[1:16]
         clust2 = HCA_thal_cell$order[17:nrow(input_thal)]
         tthal_df_ref[clust1,]$cluster = 1
         tthal_df_ref[clust2,]$cluster = 2
         ggplot(tthal_df_ref, aes(damgoEffect)) +
             geom_histogram(data=subset(tthal_df_ref,cluster==1), alpha=0.5, fill='red', binwi
             geom_histogram(data=subset(tthal_df_ref,cluster==2), alpha=0.5, fill='blue',  binw
             ylim(0,10)
         print('Warning, if alpha in the above plot is not allowed, this will stall the plotti
         print('please remove the alpha=0.5 from the above ggplot section')
         clust1 = HCA_thal_cell$order[1:16]
```

```
        clust2 = HCA_thal_cell$order[17:nrow(input_thal)]
        thal_clust1 = tthal_df_ref$damgoEffect[clust1]
        thal_clust2 = tthal_df_ref$damgoEffect[clust2]
        wilcox.test(x=thal_clust1,
                    y=thal_clust2,
                    paired=FALSE)
```

```
Warning message:
Removed 1 rows containing non-finite values (stat_bin).Warning message in grid.Call.graphics(C
semi-transparency is not supported on this device: reported only once per page

[1] "Warning, if alpha in the above plot is not allowed, this will stall the plotting and subs
[1] "please remove the alpha=0.5 from the above ggplot section"


Warning message in wilcox.test.default(x = thal_clust1, y = thal_clust2, paired = FALSE):
cannot compute exact p-value with ties


Wilcoxon rank sum test with continuity correction

data:  thal_clust1 and thal_clust2
W = 181, p-value = 0.9884833
alternative hypothesis: true location shift is not equal to 0
```


count
10.0
7.5
5.0
2.5
0.0
0
25
50
75
100
damgoEffect