Data Visualization in R - basic examples

## Learning Objectives

* Learn basic chart types using ggplot2 that are appropriate for common purposes
  + condition means (bar plot, line plot)
  + condition means with a grouping variable (barplot, line plot)
  + relation between 2 variables
  + relation between 2 variables with a grouping variable
  + *challenge section* - alternative methods to visualize variability
* Learn strategies for visualizing variability
* By going through examples, develop an beginner’s understanding of the “grammar of graphics” [(Wickham, 2007)](https://doi.org/10.1198/jcgs.2009.07098) used by ggplot2 (geom, coordinate system, aesthetic component mapping, statistical function, layers, …)

### Starting off notes

Today we are going to jump straight into plotting with the ggplot2 package. The ggplot2 package will most likely not seem intuitive at first, but after a few examples you might start to get the hang of it. Andy Field’s “discovr” tutorial #3 is highly recommended for getting more comfortable with ggplot2 capabilities.  
For learning more about ggplot2 you can go through the [official documentation (the cheat sheet is a good resource)](https://ggplot2.tidyverse.org/) and [examples in the R-Graph-Gallery site](https://www.r-graph-gallery.com/ggplot2-package.html).

This activity borrows the from the [PsyTeachR unit on data visualization](https://psyteachr.github.io/ug2-practical/visualisation-through-ggplot2.html) developed at the University of Glasgow [School of Psychology](https://www.gla.ac.uk/schools/psychology/) and [Institute of Neuroscience and Psychology](https://www.gla.ac.uk/researchinstitutes/neurosciencepsychology/).

## Step 1 - Get organized

* make a folder for today’s activity, with a new Rproj file
* make a “data” folder
  + Download the data file [mentalrotation.csv](../data/mentalrotation.csv) and place it in the new “data” folder
* make a “r\_docs” folder
  + start a new R Markdown doc and save it in your r\_docs folder

## Step 2 - Import data and check it out

* This data is from [Ganis and Kievit (2016)](https://doi.org/10.5334/jopd.ai), a replication of the [Shephard and Metzler (1971)](https://doi.org/10.1126/science.171.3972.701).
* In this study subjects had to mentally rotate a 3D shape and respond whether it was the same or different compared to a reference shape. The angle of rotation was manipulated (within subjects) at 0, 50, 100, and 150 degrees as well as the desired response (whether the shape was actually same or different).
* Each line in the MentalRotationBehavioralData.csv file represents 1 trial. The Time column is response time in milliseconds. Missed responses are coded as NA in the Time column and “[N/A]” in the ActualResponse column.
* **What to do first:** Make a new code chunk and use readr::read\_csv() to read in the data. Make sure that NA values are handled the way you want (click on the tibble in the Environment window pane to take a quick look).
* **What to do next:** make sure the columns that contain nominal vals are treated as nominal, using forcats::as\_factor() *take a look at the solution to see how*

Show/Hide Solution

#first import the data  
mrot\_tib <- readr::read\_csv("data/mentalrotation.csv", na = "NA")  
  
# now make sure the columns we want as factors are treated that way, using forcats::as\_factor() - we could let "Angle" be a Ratio variable   
mrot\_tib <- mrot\_tib %>% dplyr::mutate(  
 DesiredResponse = forcats::as\_factor(DesiredResponse),  
 ActualResponse = forcats::as\_factor(ActualResponse),  
 CorrectResponse = forcats::as\_factor(CorrectResponse),  
 Sex = forcats::as\_factor(Sex),  
)

* **Now skip some things you would normally do:** If you want to take a look at descriptives and distribution of response time, click the button:

Show/Hide Extra Stuff

#first import the data  
mrot\_tib <- readr::read\_csv("data/mentalrotation.csv", na = "NA")  
  
# now make sure the columns we want as nominal vals are treated as nominal, using forcats::as\_factor()  
mrot\_tib <- mrot\_tib %>% dplyr::mutate(  
 DesiredResponse = forcats::as\_factor(DesiredResponse),  
 ActualResponse = forcats::as\_factor(ActualResponse),  
 CorrectResponse = forcats::as\_factor(CorrectResponse),  
 Sex = forcats::as\_factor(Sex),  
)  
  
# first group by Participant and store in a new tibble  
mrot\_bysub <- mrot\_tib %>% drop\_na(Time) %>%   
 group\_by(Participant) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
# then average across Participants  
mrot\_bysub %>% dplyr::summarise(  
 meanRT = mean(sub\_meanRT),  
 ci.low = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymin,  
 ci.upp = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymax,  
 median = median(sub\_meanRT),  
 sd = sd(sub\_meanRT),  
 cases = n() - sum(is.na(sub\_meanRT))  
) %>%   
 knitr::kable(caption = "Response Time Descriptives", digits = 3) %>%   
 kableExtra::kable\_styling(full\_width = FALSE)  
  
# now visualize the distribution  
p1 <- mrot\_bysub %>% drop\_na(sub\_meanRT) %>%  
 ggplot( aes(x=sub\_meanRT)) + geom\_histogram(binwidth=100) + theme\_classic() +  
 labs (title = "Response Time distribution")  
p2 <- mrot\_bysub %>% drop\_na(sub\_meanRT) %>%  
 ggplot( aes(y=sub\_meanRT)) + geom\_boxplot() + theme\_classic() +   
 labs (title = "Response Time box plot")  
p3 <- mrot\_bysub %>% drop\_na(sub\_meanRT) %>%  
 ggplot( aes(sample=sub\_meanRT)) + geom\_qq() + geom\_qq\_line() + theme\_classic() +  
 labs (title = "Response Time Q-Q")  
p1  
p2  
p3  
mrot\_bysub %>% {shapiro.test(.$sub\_meanRT)}

Response Time Descriptives

meanRT

ci.low

ci.upp

median

sd

cases

2734.102

2544.947

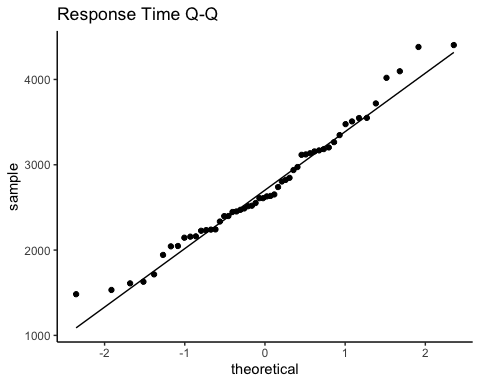
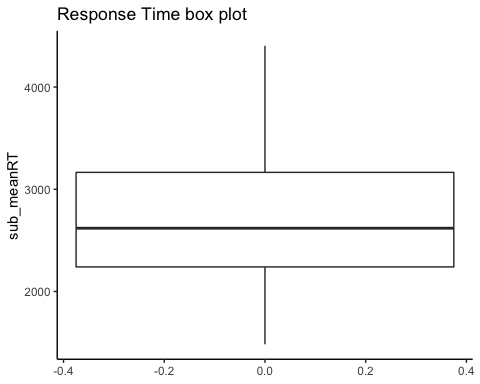
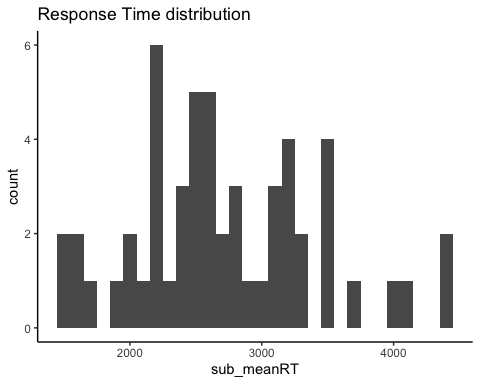
2923.256

2618.95

693.007

54

##   
## Shapiro-Wilk normality test  
##   
## data: .$sub\_meanRT  
## W = 0.97293, p-value = 0.2588



## Step 3 - make a bar plot of response time means by Angle condition

* The x-axis is the “Angle” condition (1,50,100,150), and the y-axis is the response time mean. For this exercise let’s first calculate means across trials for each subject, then calculate a sample mean across subjects, with sd based on the subject (by condition) means.
* We will only use response times from correct response trials (CorrectResponse==“Correct”)
* For most of these plots we recommend you reveal the solution immediately, then paste it into a code chunk in your own Rmd and check that it works (e.g., make sure the variable name matches for where you imported the data). Then you can look through the code and think about what each piece is for.

Show/Hide Solution

# A. first group data by Participant and Angle and store in a new tibble  
mrot\_bysub <- mrot\_tib %>% filter(CorrectResponse=="Correct") %>%   
 group\_by(Participant, Angle) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
mrot\_bysub %>% glimpse() %>%   
 knitr::kable(caption = "Participant-level Mean RT", digits = 3) %>%   
 kableExtra::kable\_styling(full\_width = FALSE)  
  
# B. then group data by Angle only to average across Participants  
mrot\_summary <- mrot\_bysub %>%   
 group\_by(Angle) %>%   
 dplyr::summarise(  
 meanRT = mean(sub\_meanRT),  
 ci.low = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymin,  
 ci.upp = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymax,  
 )  
mrot\_summary %>%   
 knitr::kable(caption = "Group-level Mean RT", digits = 3) %>%   
 kableExtra::kable\_styling(full\_width = FALSE)  
  
# C. now we can make the plot  
p1 <- mrot\_summary %>%   
 ggplot(aes(x = Angle, y = meanRT)) +   
 geom\_bar(stat = "identity") +  
 coord\_cartesian(ylim = c(0, 4500)) +   
 theme\_classic() + labs(title="Correct trial response time means", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

## Rows: 216  
## Columns: 3  
## Groups: Participant [54]  
## $ Participant <chr> "sub1", "sub1", "sub1", "sub1", "sub10", "sub10", "sub10"…  
## $ Angle <dbl> 0, 50, 100, 150, 0, 50, 100, 150, 0, 50, 100, 150, 0, 50,…  
## $ sub\_meanRT <dbl> 1512.125, 1976.750, 2771.300, 2931.143, 1939.333, 3043.57…

Participant-level Mean RT

Participant

Angle

sub\_meanRT

sub1

0

1512.125

sub1

50

1976.750

sub1

100

2771.300

sub1

150

2931.143

sub10

0

1939.333

sub10

50

3043.571

sub10

100

4740.059

sub10

150

3927.462

sub11

0

1808.417

sub11

50

2688.625

sub11

100

3837.409

sub11

150

4225.053

sub12

0

1541.455

sub12

50

1871.917

sub12

100

2571.182

sub12

150

2687.235

sub13

0

920.500

sub13

50

1196.478

sub13

100

1650.696

sub13

150

2016.667

sub14

0

1247.208

sub14

50

1522.435

sub14

100

1840.667

sub14

150

1755.136

sub15

0

1898.792

sub15

50

1921.909

sub15

100

2683.500

sub15

150

2888.857

sub16

0

2052.625

sub16

50

2482.636

sub16

100

3243.667

sub16

150

3343.913

sub17

0

1604.261

sub17

50

2088.696

sub17

100

2893.773

sub17

150

3395.750

sub18

0

2790.364

sub18

50

3863.476

sub18

100

4644.722

sub18

150

4923.267

sub19

0

3546.304

sub19

50

4714.647

sub19

100

4918.200

sub19

150

4564.533

sub2

0

1251.042

sub2

50

1779.565

sub2

100

2267.857

sub2

150

2847.381

sub20

0

1384.913

sub20

50

2037.667

sub20

100

2489.955

sub20

150

2830.905

sub21

0

1231.217

sub21

50

1558.263

sub21

100

1737.750

sub21

150

1694.571

sub22

0

2361.217

sub22

50

3691.174

sub22

100

4090.773

sub22

150

4621.900

sub23

0

1558.478

sub23

50

1892.632

sub23

100

2379.368

sub23

150

2491.400

sub24

0

1789.522

sub24

50

2194.250

sub24

100

2714.286

sub24

150

3479.579

sub25

0

1188.571

sub25

50

1408.609

sub25

100

2182.895

sub25

150

2040.000

sub26

0

1514.609

sub26

50

1910.458

sub26

100

2551.364

sub26

150

2957.182

sub27

0

1982.000

sub27

50

2591.364

sub27

100

2646.333

sub27

150

2951.333

sub28

0

1679.958

sub28

50

2327.636

sub28

100

2920.174

sub28

150

2910.619

sub29

0

2106.389

sub29

50

2571.818

sub29

100

3421.579

sub29

150

3261.238

sub3

0

1688.591

sub3

50

1862.375

sub3

100

2540.737

sub3

150

2539.526

sub30

0

1991.478

sub30

50

2645.696

sub30

100

3992.682

sub30

150

4420.789

sub31

0

1989.292

sub31

50

2985.286

sub31

100

4298.444

sub31

150

4660.118

sub32

0

2236.208

sub32

50

2331.000

sub32

100

2947.957

sub32

150

2826.905

sub33

0

1453.682

sub33

50

1897.273

sub33

100

1920.895

sub33

150

2643.762

sub34

0

2038.870

sub34

50

3014.000

sub34

100

3621.455

sub34

150

3895.905

sub35

0

2096.083

sub35

50

3041.083

sub35

100

3287.565

sub35

150

3509.273

sub36

0

1570.042

sub36

50

2022.091

sub36

100

2756.458

sub36

150

3232.667

sub37

0

1727.583

sub37

50

2543.318

sub37

100

2536.158

sub37

150

2865.190

sub38

0

2383.292

sub38

50

3134.333

sub38

100

3976.591

sub38

150

3891.000

sub39

0

2089.130

sub39

50

2594.684

sub39

100

2807.850

sub39

150

2913.308

sub4

0

1500.583

sub4

50

2032.435

sub4

100

2258.870

sub4

150

2353.091

sub40

0

2112.750

sub40

50

3019.348

sub40

100

3346.909

sub40

150

4004.789

sub41

0

1524.833

sub41

50

2359.458

sub41

100

2870.591

sub41

150

3623.900

sub42

0

2188.458

sub42

50

1940.545

sub42

100

2920.682

sub42

150

3365.091

sub43

0

1914.833

sub43

50

2827.000

sub43

100

3541.312

sub43

150

4030.600

sub44

0

3151.364

sub44

50

4432.273

sub44

100

4944.609

sub44

150

4959.312

sub45

0

1898.292

sub45

50

2616.739

sub45

100

4288.286

sub45

150

4063.833

sub46

0

2441.571

sub46

50

3821.957

sub46

100

3929.300

sub46

150

4107.619

sub47

0

1720.391

sub47

50

2464.864

sub47

100

3462.263

sub47

150

3599.684

sub48

0

1547.833

sub48

50

2397.500

sub48

100

3166.522

sub48

150

3210.947

sub49

0

2003.042

sub49

50

2409.375

sub49

100

2824.429

sub49

150

3371.583

sub5

0

2133.957

sub5

50

2513.409

sub5

100

2560.190

sub5

150

2649.000

sub50

0

2651.095

sub50

50

3045.522

sub50

100

3508.375

sub50

150

3179.762

sub51

0

1444.158

sub51

50

2062.652

sub51

100

2508.882

sub51

150

2475.611

sub52

0

3539.786

sub52

50

3006.048

sub52

100

3529.818

sub52

150

3542.913

sub53

0

3226.818

sub53

50

4078.952

sub53

100

4435.143

sub53

150

4308.944

sub54

0

2299.286

sub54

50

2866.478

sub54

100

3475.800

sub54

150

3566.050

sub6

0

1307.522

sub6

50

1376.478

sub6

100

1558.273

sub6

150

1868.870

sub7

0

2029.304

sub7

50

2952.304

sub7

100

2892.958

sub7

150

3086.136

sub8

0

1453.174

sub8

50

1824.125

sub8

100

2829.042

sub8

150

2776.435

sub9

0

2012.125

sub9

50

2523.348

sub9

100

2945.348

sub9

150

3545.619

Group-level Mean RT

Angle

meanRT

ci.low

ci.upp

0

1931.013

1777.462

2084.565

50

2518.084

2312.799

2723.369

100

3105.220

2877.041

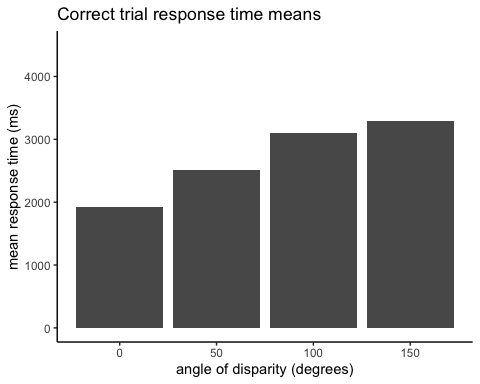
3333.400

150

3293.210

3075.000

3511.421



##### A few key elements in the code:

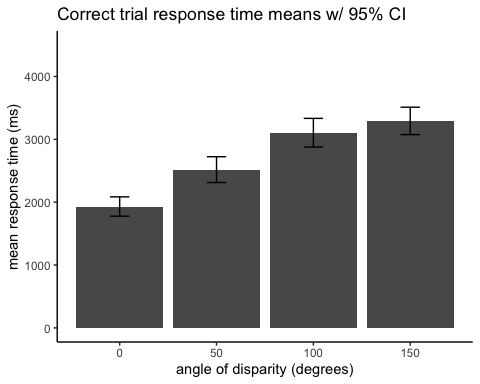
1. first we compute participant means for each Angle condition. the filter(CorrectResponse=="Correct") makes sure we only use trials where the response was correct.
2. the plotting command is all in the block of code containing the ggplot() function - mrot\_summary %>% sends the summary stats to the ggplot() functions.
3. ggplot(aes(x = Angle, y = meanRT)) sets the “Angle” column of the data to the x-axis and the “meanRT” column to the y-axis. These are called “aesthetics” (hence the aes() function) in ggplot terminology.
4. After the main ggplot() function we add “layers”. The main layer is the bars that are created by geom\_bar(). We specify stat = "identity" because the statistic we want to plot is already calculated in the mrot\_summary tibble the we sent as input to ggplot(). If instead we sent a tibble of participant-wise means, we could specify stat = "mean" to have ggplot compute the group means.
5. coord\_cartesian(ylim = c(0, 4500)) controls the range of the y-axis (we could add xlim = if we wanted to). labs() controls the labels. theme\_classic is one of many options for themes that specify color, font, positioning, etc.

#### Error bars

* Okay, so we can see the means, but there’s no information at all about variability, so let’s at least put error bars on the bar plot.
* Let’s set them to equal to the 95% confidence interval around the mean. The solution above already calculated ci.low and ci.upp, so just add a layer to the ggplot() code including this geom\_errorbar(aes(ymin=ci.low, ymax=ci.upp), width=10) - the ymin/ymax are values for the bars, and the width is in units of the x-axis (which is 50 units between levels because “Angle” is treated as an interval measure rather than as a factor).

Show/Hide Solution

# add ymin and ymax to the aesthetics list, which are needed for geom\_errorbar  
p1 <- mrot\_summary %>%   
 ggplot(aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp)) +   
 geom\_bar(stat = "identity") +  
 geom\_errorbar(width=10) +  
 coord\_cartesian(ylim = c(0, 4500)) +   
 theme\_classic() + labs(title="Correct trial response time means w/ 95% CI", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

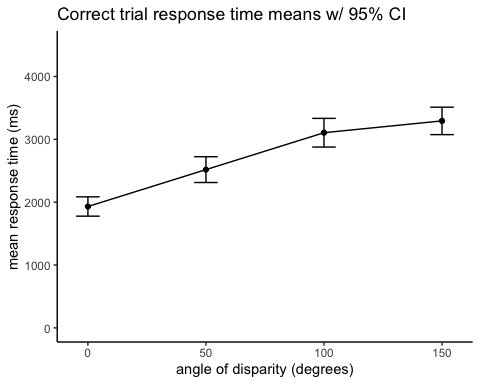


#### Line Plot

* In some cases you might prefer a point and line plot instead of bars, though it provides the same amount of information here. Let’s see how it looks as a line plot instead.
* Instead of geom\_bar use geom\_point to get a point at each mean.
* Now add another layer with geom\_line(aes(group=1),stat = "identity") to draw lines connecting the points (the group=1 part specifies there is only one group of lines).

Show/Hide Solution

p1 <- mrot\_summary %>%   
 ggplot(aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp)) +   
 geom\_point(stat = "identity") +  
 geom\_line(aes(group=1),stat = "identity") +  
 geom\_errorbar(width=10) +  
 coord\_cartesian(ylim = c(0, 4500)) +   
 theme\_classic() + labs(title="Correct trial response time means w/ 95% CI", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

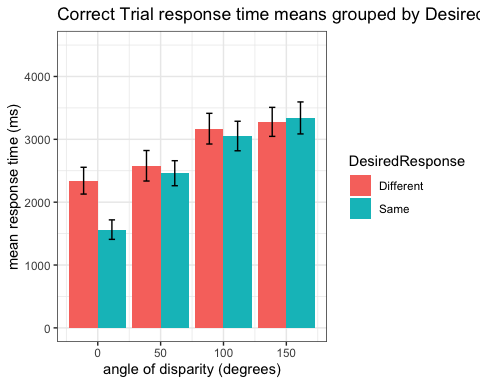


## Step 4 - group the bar plot by “DesiredResponse”

* The column “DesiredResponse” indicates whether the target shape was the same or different than the reference shape. Let’s treat the data like a 2x4 design and split each bar into two (one bar for “same” and one for “different”). We’ll color code the “DesiredResponse” values.
* to accomplish this, first we have to re-do the grouping to also group by “DesiredResponse” values
* then we add a new aesthetic called “fill” (referring to the bar fill color) to designate the colors by DesiredResponse values
* then we need to adjust the position of the bars so that they are not on top of each other - use position=position\_dodge() in the geoms
* Copy the Solution code below and then look at it piece by piece to understand it

Show/Hide Solution

# regroup data to include additional factor  
mrot\_bysub <- mrot\_tib %>% filter(CorrectResponse=="Correct") %>%  
 group\_by(Participant, Angle, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
mrot\_summary <- mrot\_bysub %>%   
 group\_by(Angle,DesiredResponse) %>%   
 dplyr::summarise(  
 meanRT = mean(sub\_meanRT),  
 ci.low = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymin,  
 ci.upp = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymax,  
 )  
# add fill aesthetic as DesiredResponse  
# use position\_dodge to move the bars and errorbars horizontally so   
# they aren't on top of each other. A dodge value of .9\*(distance between levels) is generally a good dodge value.  
p1 <- mrot\_summary %>%   
 ggplot(aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp, fill=DesiredResponse)) +   
 geom\_bar(stat = "identity", position=position\_dodge(45)) +  
 geom\_errorbar(width=10, position=position\_dodge(45)) +  
 coord\_cartesian(ylim = c(0, 4500)) +   
 theme\_bw() + labs(title="Correct Trial response time means grouped by DesiredResponse", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

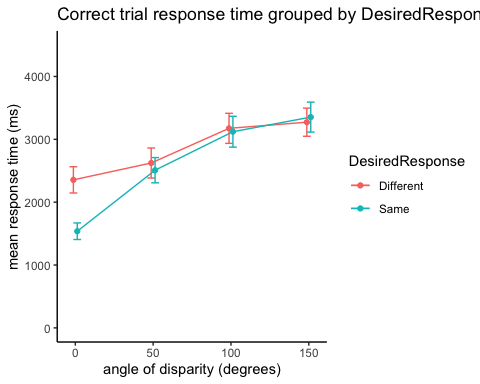


#### Line Plot, grouped by “DesiredResponse”

* Cool. Now that you’ve figured that out, try the same thing but as a line plot.
* Instead of fill, use the color aesthetic.
* Use a smaller dodge value (or no dodge value at all) now that you don’t have thick bars. If you specify a dodge value, you’ll need to do so for each of the three geom elements (geom\_point, geom\_line, and geom\_errorbar)

Show/Hide Solution

mrot\_bysub <- mrot\_tib %>% drop\_na(Time) %>%  
 group\_by(Participant, Angle, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
mrot\_summary <- mrot\_bysub %>%   
 group\_by(Angle,DesiredResponse) %>%   
 dplyr::summarise(  
 meanRT = mean(sub\_meanRT),  
 ci.low = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymin,  
 ci.upp = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymax,  
 )  
p1 <- mrot\_summary %>%   
 ggplot(aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp, color=DesiredResponse)) +   
 geom\_point(stat = "identity", position=position\_dodge(5)) +  
 geom\_line(aes(group=DesiredResponse), position=position\_dodge(5)) +  
 geom\_errorbar(width=10, position=position\_dodge(5)) +  
 coord\_cartesian(ylim = c(0, 4500)) +   
 theme\_classic() + labs(title="Correct trial response time grouped by DesiredResponse", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1



## Step 5 - Visualize a relation between two variables

* Is reaction time related to accuracy?
* Let’s compute %correct for each subject, by counting the proportion of “Correct” values in the “CorrectResponse” column. We exclude trials with no response.

Show/Hide Solution

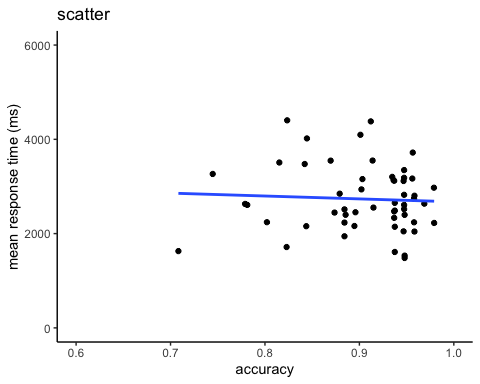
mrot\_bysub <- mrot\_tib %>% drop\_na() %>%   
 group\_by(Participant) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time),  
 sub\_accuracy = sum(CorrectResponse=="Correct")/n()  
 )

#### Scatter plot of response time by accuracy

* now we can specify the x and y axes, and use geom\_point() to plot a point for each subject with accuracy on the x-axis and response time on the y-axis
* we can add a regression line by adding a layer with stat\_smooth(method = "lm", formula = "y ~ x")

Show/Hide Solution

p1 <- mrot\_bysub %>%   
 ggplot(aes(x = sub\_accuracy, y = sub\_meanRT)) +   
 geom\_point(stat="identity") +  
 stat\_smooth(method = "lm", formula = "y ~ x", se = FALSE) + #intercept automatically included  
 coord\_cartesian(ylim = c(0, 6000), xlim = c(.6,1)) +   
 theme\_classic() + labs(title="scatter", y = "mean response time (ms)", x = "accuracy")  
p1



#### Scatter plot of response time by accuracy, grouped by DesiredResponse

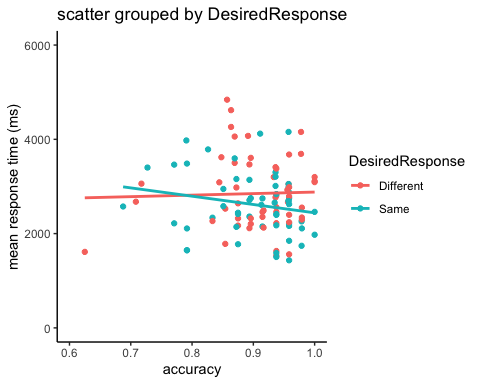
* What if we wanted to split up subject RT and accuracy by DesiredResponse (whether the shapes were same or different)?
* just (a) recompute the subject-level means, grouping by DesiredResponse, and (b) add a color mapping aesthetic color=DesiredResponse in the aes() part of the ggplot code

Show/Hide Solution

mrot\_bysub <- mrot\_tib %>% drop\_na() %>%   
 group\_by(Participant, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time),  
 sub\_accuracy = sum(CorrectResponse=="Correct")/n()  
 )

## `summarise()` has grouped output by 'Participant'. You can override using the `.groups` argument.

p1 <- mrot\_bysub %>%   
 ggplot(aes(x = sub\_accuracy, y = sub\_meanRT, color = DesiredResponse)) +   
 geom\_point(stat="identity") +  
 stat\_smooth(method = "lm", formula = "y ~ x", se = FALSE) + #intercept automatically included  
 coord\_cartesian(ylim = c(0, 6000), xlim = c(.6,1)) +   
 theme\_classic() + labs(title="scatter grouped by DesiredResponse", y = "mean response time (ms)", x = "accuracy")  
p1



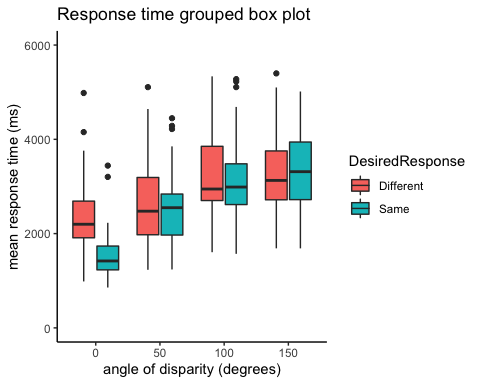
## Challenge: For your reference (or if you have extra time in lab) - here are some alternative approaches to visualizing variability:

#### 1. Use a box plot instead of plotting means:

* notice that we have to re-specify the x-axis (Angle) variable as a factor to get geom\_boxplot to work correctly mutate(Angle=forcats::as\_factor(Angle))
* notice that we pass in subject level means, instead of group means
* also notice that we have to expand the ylim range to see all outliers

Show/Hide Code and Plot

# regroup data   
mrot\_bysub <- mrot\_tib %>% mutate(Angle=forcats::as\_factor(Angle)) %>% drop\_na(Time) %>%   
 group\_by(Participant, Angle, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
  
p1 <- mrot\_bysub %>%   
 ggplot(aes(x = Angle, y = sub\_meanRT, fill=DesiredResponse)) +   
 geom\_boxplot() +  
 coord\_cartesian(ylim = c(0, 6000)) +   
 theme\_classic() + labs(title="Response time grouped box plot", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

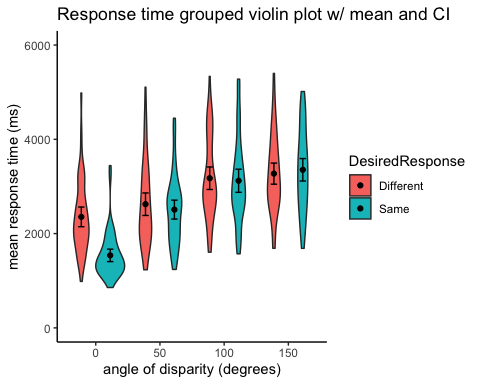


#### 2. Use a violin plot layer under plotted means

* again, we pass in subject level means
* again, we use an expanded ylim range
* for geom\_point() and geom\_errorbar this time we specify the group summarized data

Show/Hide Code and Plot

# group data   
mrot\_bysub <- mrot\_tib %>% mutate(Angle=forcats::as\_factor(Angle)) %>% drop\_na(Time) %>%   
 group\_by(Participant, Angle, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
mrot\_summary <- mrot\_bysub %>%   
 group\_by(Angle,DesiredResponse) %>%   
 dplyr::summarise(  
 meanRT = mean(sub\_meanRT),  
 ci.low = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymin,  
 ci.upp = ggplot2::mean\_cl\_normal(sub\_meanRT)$ymax,  
 )  
p1 <- mrot\_bysub %>%   
 ggplot(aes(x = Angle, y = sub\_meanRT, fill=DesiredResponse)) +   
 geom\_violin(position = position\_dodge(.9)) +  
 geom\_point(data=mrot\_summary, aes(x = Angle, y = meanRT), stat="identity", position = position\_dodge(.9)) +  
 geom\_errorbar(data=mrot\_summary, aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp), stat="identity", width=.2, position\_dodge(.9)) +  
 coord\_cartesian(ylim = c(0, 6000)) +   
 theme\_classic() + labs(title="Response time grouped violin plot w/ mean and CI", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1



#### 3. Show all the points: Use a 1D scatter with mean and error bars on top

* again, we pass in subject level means
* again, we use an expanded ylim range
* we do the mean and error bars just like in the previous example using group-summarized data

Show/Hide Code and Plot

# regroup data   
mrot\_bysub <- mrot\_tib %>% mutate(Angle=forcats::as\_factor(Angle)) %>% drop\_na(Time) %>%   
 group\_by(Participant, Angle, DesiredResponse) %>%   
 dplyr::summarise(  
 sub\_meanRT = mean(Time)  
 )  
p1 <- mrot\_bysub %>%   
 ggplot(aes(x = Angle, y = sub\_meanRT, color=DesiredResponse, shape=DesiredResponse)) +   
 geom\_jitter(position=position\_jitterdodge(.6)) +  
 geom\_point(data=mrot\_summary, aes(x = Angle, y = meanRT),color="black", stat="identity", position = position\_dodge(.75), show.legend = FALSE) +  
 geom\_errorbar(data=mrot\_summary, aes(x = Angle, y = meanRT, ymin=ci.low, ymax=ci.upp), color="black", stat="identity", width=.2, position\_dodge(.75), show.legend = FALSE) +  
 coord\_cartesian(ylim = c(0, 6000)) +   
 theme\_classic() + labs(title="Response time grouped 1D scatter plot", y = "mean response time (ms)", x = "angle of disparity (degrees)")  
p1

