RepGreene_Data_analysis

lynde

2022-12-07

##Libaries libraries

##Motivation

We're trying to replicate the main finding which will involve running a series of linear models and ultimately a linear mixed effect model. First will be looking at effect of load on judgement and we shouldn't see significance. Then effect of load on utilitarian vs deontological choice each. This should, if replicated, demonstrate a significant effect on utilitarian judgments only.

Original Author's findings "There was no main effect of load ($F(1, 83.2) = 2.29$, $p = .13$). There was a margin- ally significant main effect of judgment ($F(1, 71.7) = 3.9$, $p = .052$), with longer RT for utilitarian judgments (LS Means (SEM) ms: utilitarian = 6130 (207), non-utili- tarian = 5736 (221)). Critically, we observed the predicted interaction between load and judgment ($F(1, 62.9) = 8.5$, $p = .005$). (See Fig. 1.) Planned post hoc contrasts revealed a predicted increase in RT for utilitarian judgment under load ($F(1, 106.3) = 9.8$, $p = .002$; LS Means (SEM) ms: load = 6506 (238), no load = 5754 (241)), but no difference in RT for non-utilitarian judgment resulting from load ($F(1, 169.6) = .10$, $p = .75$; LS Means: load = 5691 (264), no load = 5781 (261)). Utilitarian judgments were slower than non-utilitarian judgments under load ($p = .001$), but there was no such effect in the absence of load ($p = .91$). This general pattern also held when item, rather than participant, was modeled as a random effect, although the results in this analysis were not as strong. There was no effect of load on judgment ($v2(1, N = 82) = .24$, $p = .62$), with 61% utilitarian judgments under load (95% CI: 57–66%) and 60% (95% CI: 55–64%) in the absence of load"

##Loading

```
setwd("~/F.Replications/Replications/Greene_Rep/Data")

# pilot_a_data <- read_csv("Data/pilot_a_data.csv")
# View(pilot_a_data)
#
# raw_data_pilot_a <- read_survey("pilot_a_data.csv", legacy = TRUE)
# view(raw_data_pilot_a)

raw_data <- read_survey("Greene_08_Rep_121422.csv", legacy = TRUE) %>%
  clean_names()
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character()
## )
## i Use `spec()` for the full column specifications.
```

Wringle Wrangle

Make new data frames that are tidy such that I can run the analysis more cleanly. variables I need, the choices, yes/no, the RT

The names of the variables I need to pull prolific_id vitamins /all the names of the questions (12, block 6 v 6) timing page submit

iv load for each choice and load are my predictors for "last click" marginal effects lsmean

"t1_l1_timing_vitamin_first_click", "t2_l1_timing_sub_first_click",
"t3_l1_timing_sac_first_click", "t4_l1_timing_modlb_first_click", "t5_l1_timing_lawara_first_click",
"t6_l1_timing_euthan_first_click", "t1_l2_timing_cryb_first_click", "t2_l2_timing_foot_first_click",
"t3_l2_timing_modbm_first_click", "t4_l2_timing_modsaf_first_click", "t5_l2_timing_soph_first_click",
"t6_l2_timing_vacc_first_click"

"t1_l1_vitamins", "t2_l1_sub",
"t3_l1_sac",
"t4_l1_modlb",
"t5_l1_lawara",
"t6_l1_euthan",
"t1_l2_cryb",
"t2_l2_foot",
"t3_l2_modbm",
"t4_l2_modsaf",
"t5_l2_soph",
"t6_l2_vacc",
gonna do some select and long-ing.

```
## Warning: Expected 2 pieces. Additional pieces discarded in 24 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

Got some help from Anna Xu to move the data around into where I want it. Anna noticed that the rows are every other so rather than trying to do another pivot she just kinda... moved the line around.

```
odd.ind <- seq_len(nrow(df.datalong)) %% 2
df.datalong.odds <- df.datalong[odd.ind == 1, ]
df.datalong.evens <- df.datalong[odd.ind == 0, ]

df.datalong.odds$bin.choice <- df.datalong.evens$choice

df.datalong
```

```
## # A tibble: 1,728 x 4
##    prolific_id              trial load  choice
##    <chr>                    <chr> <chr> <chr>
##  1 60bc16d1f9aef5318d50167d t1    l1    1
##  2 60bc16d1f9aef5318d50167d t1    l1    40.767
##  3 60bc16d1f9aef5318d50167d t2    l1    1
##  4 60bc16d1f9aef5318d50167d t2    l1    49.667
##  5 60bc16d1f9aef5318d50167d t3    l1    1
##  6 60bc16d1f9aef5318d50167d t3    l1    53.364
##  7 60bc16d1f9aef5318d50167d t4    l1    2
##  8 60bc16d1f9aef5318d50167d t4    l1    56.266
##  9 60bc16d1f9aef5318d50167d t5    l1    1
```

```
## 10 60bc16d1f9aef5318d50167d t5    l1    44.464
## # ... with 1,718 more rows
```

```
df.datalong.odds
```

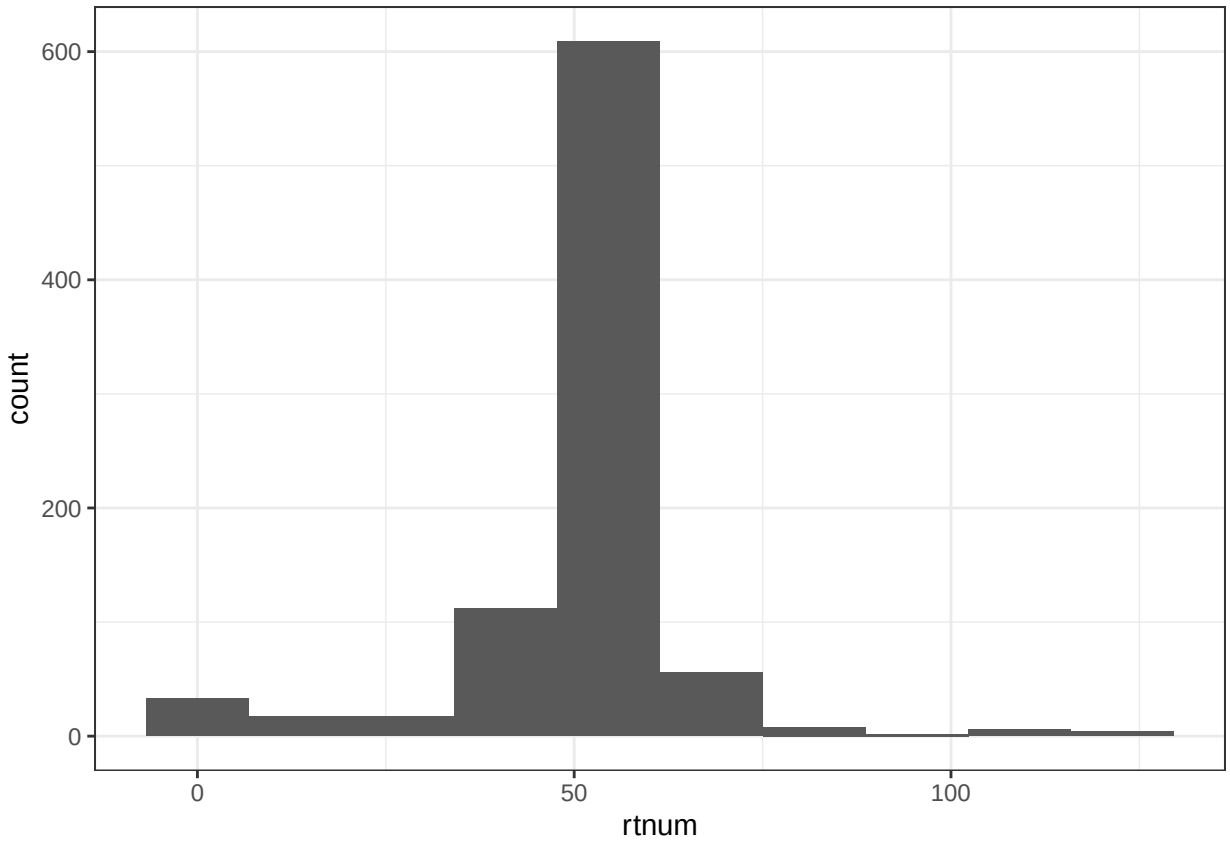
```
## # A tibble: 864 x 5
##    prolific_id              trial load  choice bin.choice
##    <chr>                    <chr> <chr> <chr>  <chr>
##  1 60bc16d1f9aef5318d50167d t1    l1    1      40.767
##  2 60bc16d1f9aef5318d50167d t2    l1    1      49.667
##  3 60bc16d1f9aef5318d50167d t3    l1    1      53.364
##  4 60bc16d1f9aef5318d50167d t4    l1    2      56.266
##  5 60bc16d1f9aef5318d50167d t5    l1    1      44.464
##  6 60bc16d1f9aef5318d50167d t6    l1    1      48.513
##  7 60bc16d1f9aef5318d50167d t1    l2    2      53.741
##  8 60bc16d1f9aef5318d50167d t2    l2    2      41.006
##  9 60bc16d1f9aef5318d50167d t3    l2    1      45.175
## 10 60bc16d1f9aef5318d50167d t4    l2    1      47.43
## # ... with 854 more rows
```

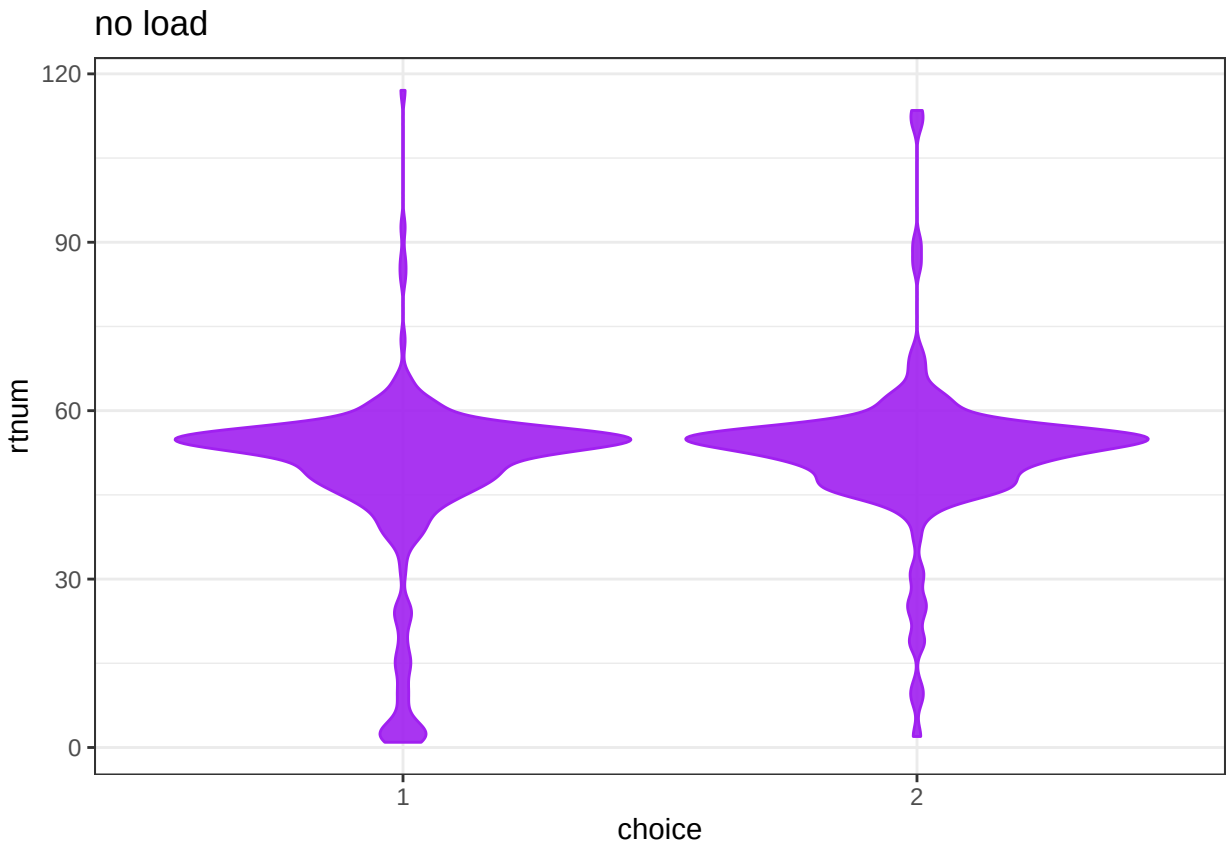
```
new.df.datalong <- df.datalong.odds

df.datalong <-new.df.datalong %>%
  rename(rt = "bin.choice") %>%
  view()
```

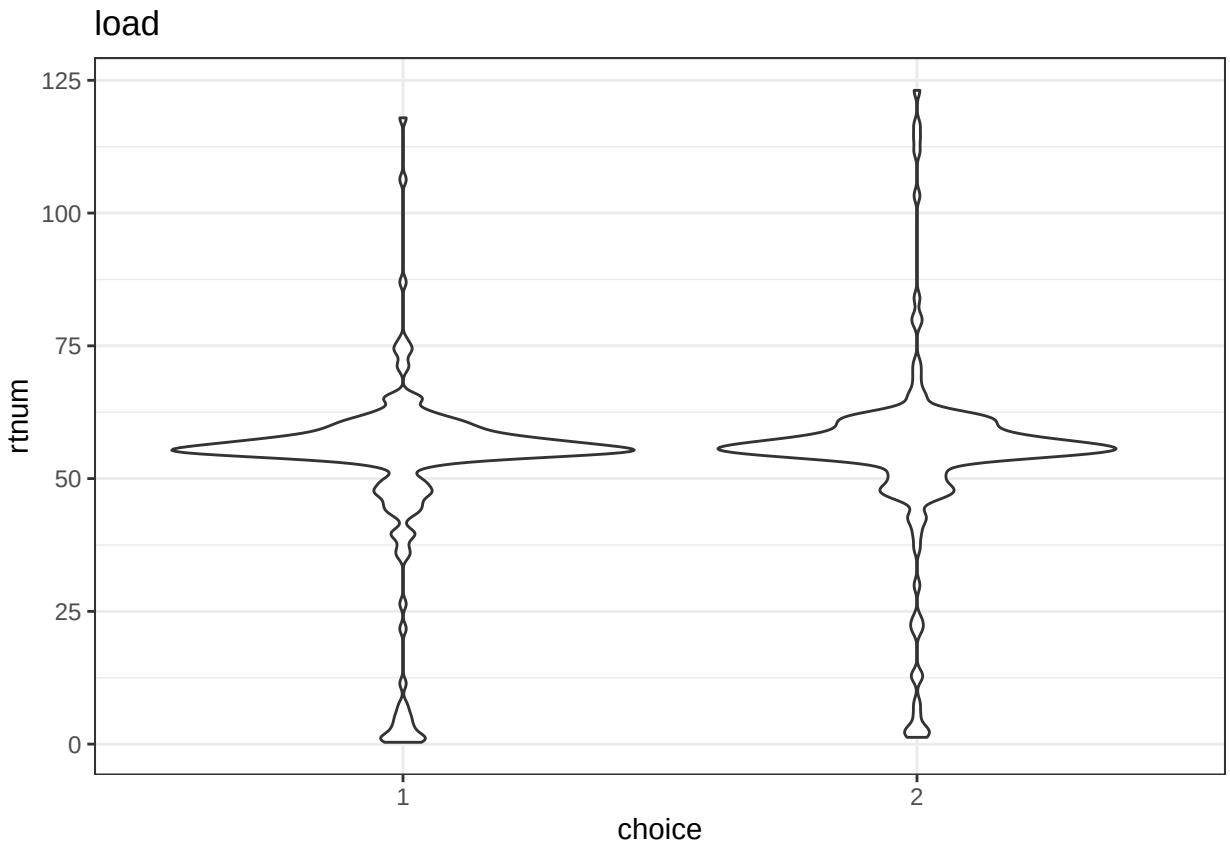
YAY long!

Run some basic descriptives?

We still need to do a little more organizing then peak at the data.


600
400
200
0
count
0
50
100
rtnum


no load
120
90
60
30
0
rtnum
1
2
choice



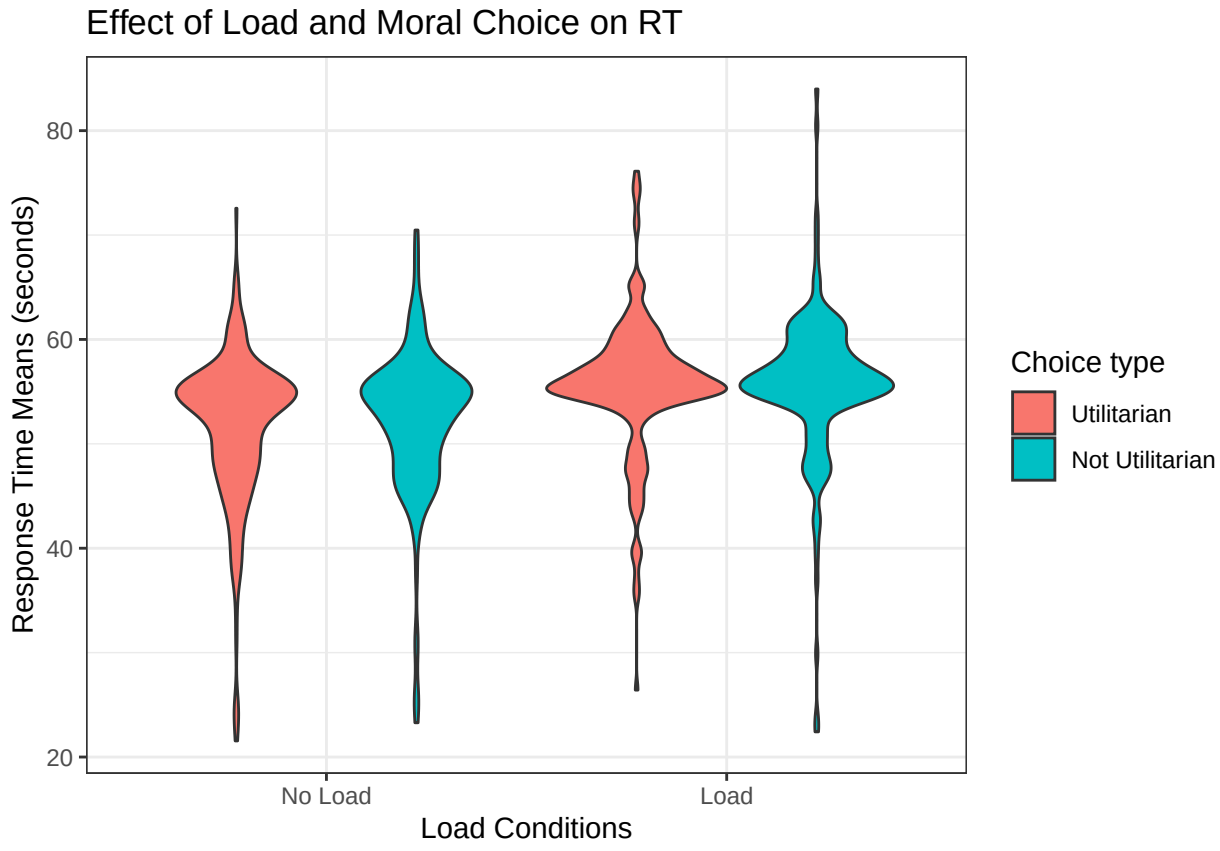
Ever notice violins look like rorschach blots?

We should probably trim the RT

From the violins above we can see that there's probably a ton of outlier RTs.

```
#trim my rts from the data & get standard devs
sumsdfdatalong <- df.datalong %>%
  group_by(load) %>%
  summarize(rtmean = mean(rtnum),
            stdrtmean = 2*sd(rtnum),
            minrt = rtmean-stdrtmean,
            maxrt = rtmean+stdrtmean)

minrt <- sumsdfdatalong$minrt
maxrt <- sumsdfdatalong$maxrt

#lets trim this to two sd like original authors
df.datalongtrim <- df.datalong %>%
  group_by(load) %>%
  filter(rtnum > minrt) %>%
  filter(rtnum < maxrt)

#making a report bc why not
#rep.dfdlongtrim <- create_report(df.datalongtrim)
#rep.dfdlong <- create_report(df.datalong)
```

```
# ^^^^ uncoment to run

#I want a good-game-plot as well
df.datalongtrim %>%
  ggplot(mapping = aes(x = load,
         y = rtnum,
         fill = choicenum
        ))+
  geom_violin()+
  labs(title = "Effect of Load and Moral Choice on RT",
       x = "Load Conditions",
       y = "Response Time Means (seconds)")+
  scale_x_discrete(labels=c("l1" = "No Load",
                            "l2" = "Load"))+
  scale_fill_discrete(name="Choice type",
                      labels=c("1" = "Utilitarian",
                               "2" = "Not Utilitarian"))
```



Model model model!

Going to do three linear models. Always doing prolific id– the participant– as a random effect.

Model 1:Choice on RT

Model 2:Load on RT

Model 3:The big kahuna! Load * choice (on RT)!

```
#modeling choice
m1<- lmer(rtnum ~choicenum +(1|prolific_id),
          data = df.datalongtrim)
report(m1)
```

```
## We fitted a linear mixed model (estimated using REML and nloptwrap optimizer) to predict rtnum with c
##
##   - The effect of choicenum [2] is statistically non-significant and positive (beta = 0.76, 95% CI [-
##
## Standardized parameters were obtained by fitting the model on a standardized version of the dataset.
```

```
summary(m1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rtnum ~ choicenum + (1 | prolific_id)
##    Data: df.datalongtrim
##
## REML criterion at convergence: 5083.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.4981 -0.4197  0.0793  0.4745  4.6719
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  prolific_id (Intercept) 27.32    5.227
##  Residual                28.19    5.309
## Number of obs: 795, groups:  prolific_id, 72
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)  53.1577     0.6719  78.8768  79.111   <2e-16 ***
## choicenum2    0.7566     0.4364 758.1595   1.734   0.0834 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##            (Intr)
## choicenum2 -0.278
```

```
#modeling load
m2 <- lmer(rtnum ~load +(1|prolific_id),
           data = df.datalongtrim)
report(m2)
```

```
## We fitted a linear mixed model (estimated using REML and nloptwrap optimizer) to predict rtnum with l
##
##   - The effect of load [12] is statistically significant and positive (beta = 3.59, 95% CI [2.89, 4.2
##
## Standardized parameters were obtained by fitting the model on a standardized version of the dataset.
```

```
summary(m2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rtnum ~ load + (1 | prolific_id)
##    Data: df.datalongtrim
##
## REML criterion at convergence: 4990.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0549 -0.3849  0.0635  0.4751  5.1037
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  prolific_id (Intercept) 27.00    5.196
##  Residual                24.85    4.985
## Number of obs: 795, groups:  prolific_id, 72
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)  51.6882     0.6627  78.9740   77.99   <2e-16 ***
## loadl2        3.5910     0.3556 721.2864   10.10   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##        (Intr)
## loadl2 -0.267
```

```
#model the load*choice and rt
m3 <- lmer(rtnum ~load *choicenum +(1|prolific_id),
           data = df.datalongtrim)
report(m3)
```

```
## We fitted a linear mixed model (estimated using REML and nloptwrap optimizer) to predict rtnum with l
##
##   - The effect of load [l2] is statistically significant and positive (beta = 4.01, 95% CI [3.07, 4.9
##   - The effect of choicenum [2] is statistically non-significant and positive (beta = 0.82, 95% CI [-
##   - The interaction effect of choicenum [2] on load [l2] is statistically non-significant and negativ
##
## Standardized parameters were obtained by fitting the model on a standardized version of the dataset.
```

```
summary(m3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rtnum ~ load * choicenum + (1 | prolific_id)
##    Data: df.datalongtrim
##
## REML criterion at convergence: 4987.2
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0437 -0.3848  0.0564  0.4657  5.0542
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  prolific_id (Intercept) 27.14    5.210
##  Residual                24.82    4.982
## Number of obs: 795, groups:  prolific_id, 72
##
## Fixed effects:
##                   Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)        51.3770     0.6974  95.3172  73.673   <2e-16 ***
## loadl2              4.0102     0.4766 721.3052   8.415   <2e-16 ***
## choicenum2          0.8204     0.5616 741.4926   1.461    0.144
## loadl2:choicenum2  -1.0490     0.7374 724.0550  -1.422    0.155
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) loadl2 chcnm2
## loadl2      -0.307
## choicenum2  -0.305  0.383
## ldl2:chcnm2  0.207 -0.660 -0.678
```

```
#looking at model comparisons
anova(m2, m3)
```

```
## refitting model(s) with ML (instead of REML)

## Data: df.datalongtrim
## Models:
## m2: rtnum ~ load + (1 | prolific_id)
## m3: rtnum ~ load * choicenum + (1 | prolific_id)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m2    4 4999.7 5018.4 -2495.8   4991.7
## m3    6 5001.2 5029.2 -2494.6   4989.2 2.4809  2     0.2893
```

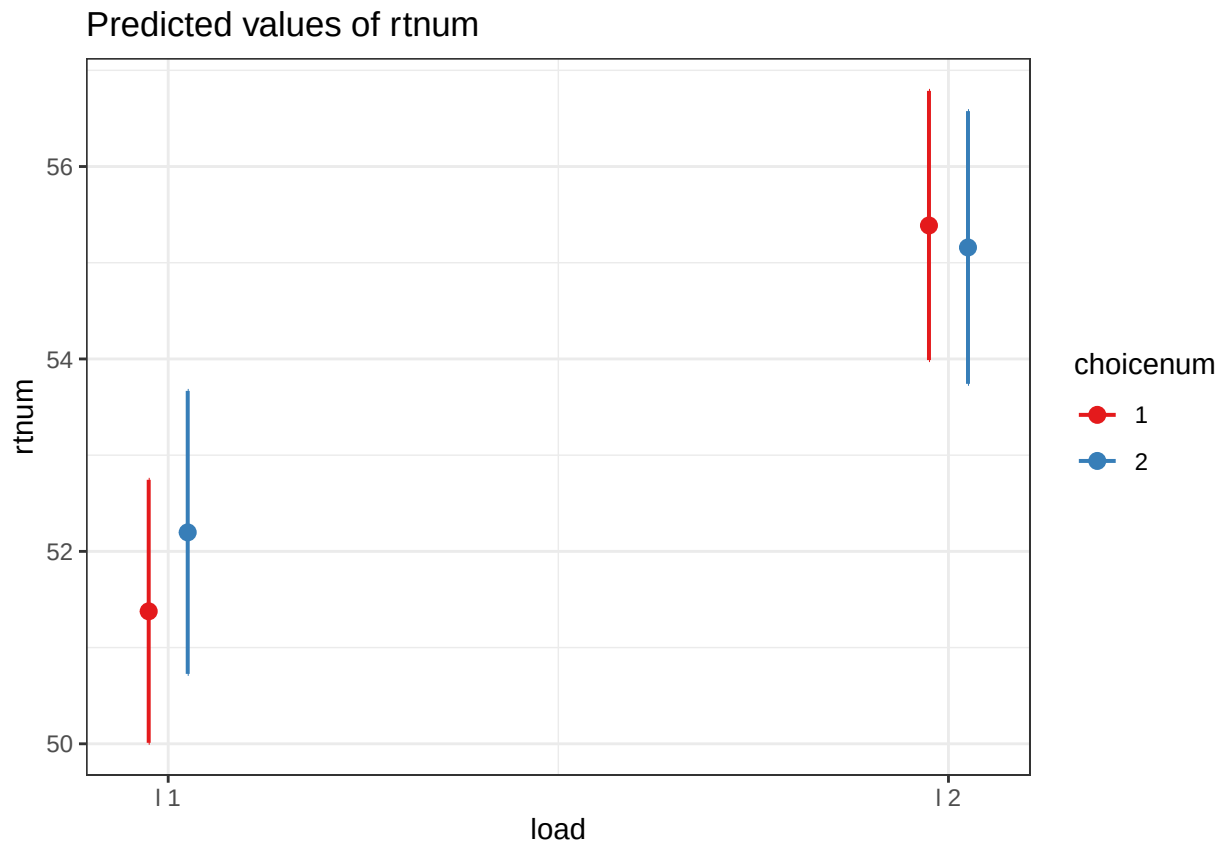
```
anova(m1, m2)
```

```
## refitting model(s) with ML (instead of REML)

## Data: df.datalongtrim
## Models:
## m1: rtnum ~ choicenum + (1 | prolific_id)
## m2: rtnum ~ load + (1 | prolific_id)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m1    4 5092.3 5111.1 -2542.2   5084.3
## m2    4 4999.7 5018.4 -2495.8   4991.7 92.693  0
```

```
anova(m1, m3)
```

```
## refitting model(s) with ML (instead of REML)

## Data: df.datalongtrim
## Models:
## m1: rtnum ~ choicenum + (1 | prolific_id)
## m3: rtnum ~ load * choicenum + (1 | prolific_id)
##    npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## m1    4 5092.3 5111.1 -2542.2   5084.3
## m3    6 5001.2 5029.2 -2494.6   4989.2 95.174  2  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#early peek at the model
plot_model(m3, type = "int")
```



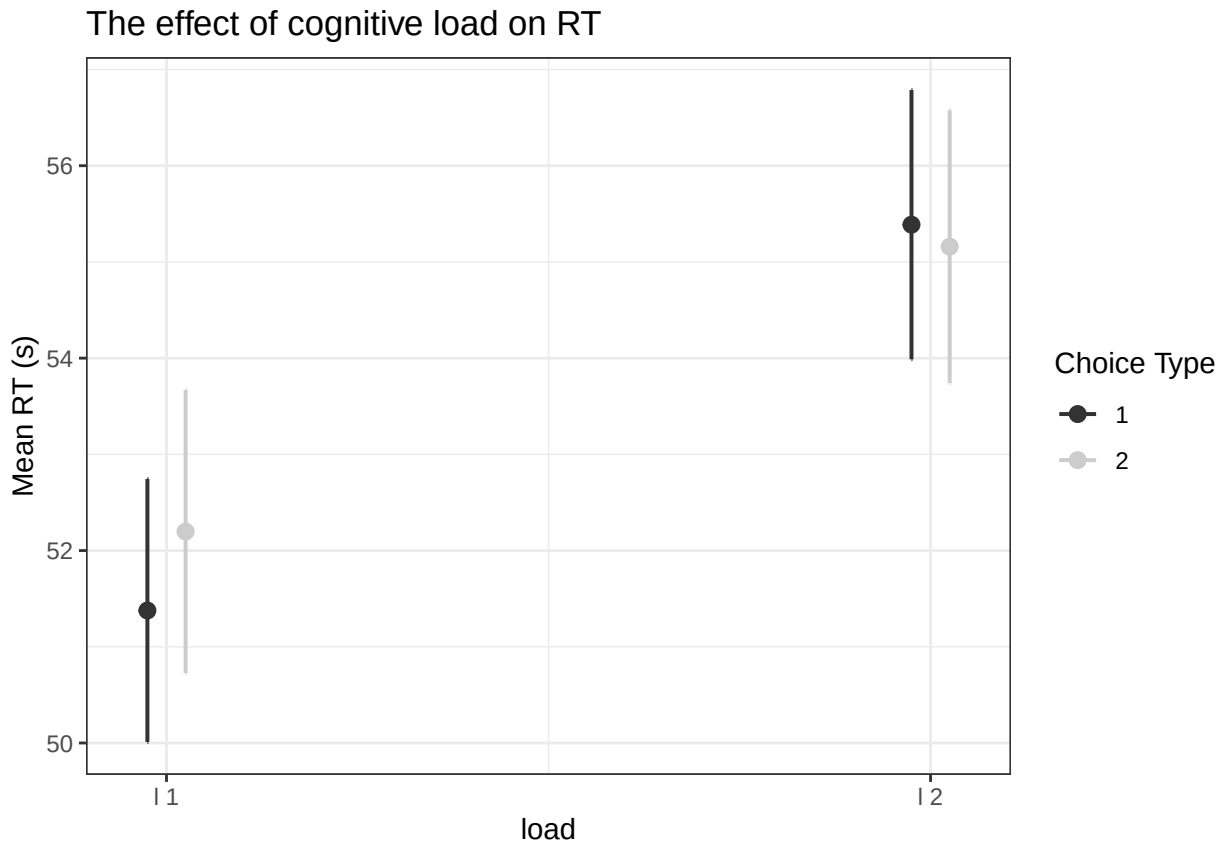

```
#save model outputs to make a figure
means <- estimate_means(m3)
```

```
## We selected `at = c("load", "choicenum")`.
```

```
as.factor(means$load) #this will make the graphing easier
```

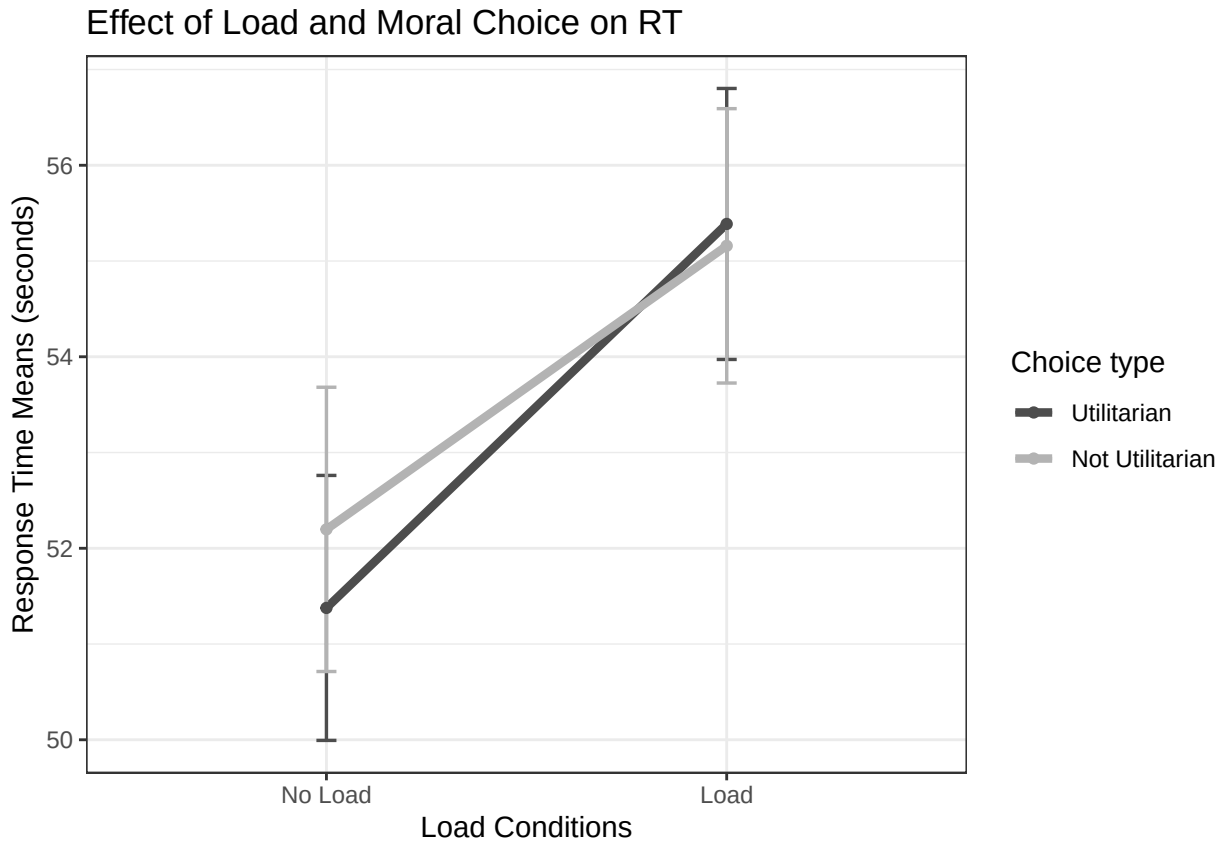
```
## [1] l1 l2 l1 l2
## Levels: l1 l2
```

```
#trying to use a different package to graph the model
plot_model(m3, type = "int",
           axis.title = "Mean RT (s)",
           colors = "gs",
           legend.title = "Choice Type",
           title = "The effect of cognitive load on RT")
```




```
#fuck it I'm gonna do ggplot
means %>%
  ggplot(mapping = aes(x = load,
                       y = Mean,
                       group = choicenum,
                       color = choicenum
                        ))+
  geom_line(size = 1.5)+
  geom_errorbar(aes(ymin=CI_low, ymax=CI_high), width=.05, size = 0.6)+
  geom_point()+
  labs(title = "Effect of Load and Moral Choice on RT",
       x = "Load Conditions",
```

```
      y = "Response Time Means (seconds)")+
 scale_x_discrete(labels=c("l1" = "No Load",
                           "l2" = "Load"))+
 scale_colour_grey(name="Choice type",
                   labels=c("1" = "Utilitarian",
                            "2" = "Not Utilitarian"),
                   start = 0.3,
                   end = 0.7
                   )
```




```
#tada!
```

Soooo this is not great huh. Seems like (a) rt significantly increases as a function of load irrespective of choice type. Frankly I'm not suprised. Further, we did not replicate the selective interference effect on utilitarian choices. Womp womp.

Time to write up the report!

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
```

```
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] esquisse_1.1.2     modelbased_0.7.1   sjmisc_2.8.9       sjPlot_2.8.12
##  [5] DataExplorer_0.8.2 reshape2_1.4.4     qualtRics_3.1.7    report_0.5.0.9000
##  [9] afex_1.0-1         forcats_0.5.1      stringr_1.4.0      dplyr_1.0.7
## [13] purrr_0.3.4        readr_2.1.1        tidyr_1.1.4        tibble_3.1.5
## [17] ggplot2_3.3.5      tidyverse_1.3.1    emmeans_1.7.2      lme4_1.1-27.1
## [21] Matrix_1.3-4       broom.mixed_0.2.7  janitor_2.1.0      knitr_1.37
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1        backports_1.2.1     plyr_1.8.6
##   [4] igraph_1.2.11       splines_4.1.1       digest_0.6.28
##   [7] htmltools_0.5.4     lmerTest_3.1-3      fansi_0.5.0
##  [10] magrittr_2.0.1      tzdb_0.1.2          openxlsx_4.2.5.1
##  [13] modelr_0.1.8        vroom_1.5.7         shinybusy_0.3.1
##  [16] colorspace_2.0-2    rvest_1.0.2         haven_2.4.3
##  [19] xfun_0.29           crayon_1.4.1        jsonlite_1.7.2
##  [22] glue_1.6.1          gtable_0.3.0        sjstats_0.18.2
##  [25] phosphoricons_0.1.2 car_3.0-12          abind_1.4-5
##  [28] scales_1.1.1        mvtnorm_1.1-3       DBI_1.1.1
##  [31] ggeffects_1.1.4     Rcpp_1.0.7          xtable_1.8-4
##  [34] performance_0.8.0   foreign_0.8-81      bit_4.0.4
##  [37] datawizard_0.3.0    htmlwidgets_1.5.4   httr_1.4.2
##  [40] RColorBrewer_1.1-2  ellipsis_0.3.2      pkgconfig_2.0.3
##  [43] farver_2.1.0        sass_0.4.4          dbplyr_2.1.1
##  [46] utf8_1.2.2          tidyselect_1.1.1    labeling_0.4.2
##  [49] rlang_1.0.6         later_1.3.0         effectsize_0.6.0.2
##  [52] munsell_0.5.0       cellranger_1.1.0    tools_4.1.1
##  [55] cachem_1.0.6        cli_3.4.1           generics_0.1.1
##  [58] sjlabelled_1.2.0    broom_0.7.9         evaluate_0.14
##  [61] fastmap_1.1.0       yaml_2.2.1          bit64_4.0.5
##  [64] fs_1.5.2            zip_2.2.0           nlme_3.1-153
##  [67] reactable_0.4.1     mime_0.12           xml2_1.3.2
##  [70] pbkrtest_0.5.1      compiler_4.1.1      rstudioapi_0.13
##  [73] datamods_1.4.0      curl_4.3.2          reprex_2.0.1
##  [76] bslib_0.4.2         stringi_1.7.5       highr_0.9
##  [79] parameters_0.17.0   lattice_0.20-45     nloptr_2.0.0
##  [82] vctrs_0.3.8         pillar_1.6.4        lifecycle_1.0.1
##  [85] networkD3_0.4       jquerylib_0.1.4     estimability_1.3
##  [88] data.table_1.14.2   insight_0.16.0      httpuv_1.6.7
##  [91] R6_2.5.1            promises_1.2.0.1    gridExtra_2.3
##  [94] rio_0.5.29          writexl_1.4.1       boot_1.3-28
##  [97] MASS_7.3-54         assertthat_0.2.1    shinyWidgets_0.7.5
## [100] withr_2.5.0         bayestestR_0.11.5   parallel_4.1.1
## [103] hms_1.1.1           grid_4.1.1          coda_0.19-4
## [106] minqa_1.2.4         rmarkdown_2.11      snakecase_0.11.0
## [109] carData_3.0-5       numDeriv_2016.8-1.1 shiny_1.7.4
## [112] lubridate_1.8.0
```