R Studio Code - Synthesis of Experiment Results

2025-03-30

```
#load in necessary packages
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── 
tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.1     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to
force all conflicts to become errors

library(ggplot2)
library(lme4)

## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

library(effects)

## Loading required package: carData
## lattice theme set by effectsTheme()
## See ?effectsTheme for details.

library(ggeffects)
library(dplyr)
library(readr)
library(car)
```

```
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(brms)
```

```
## Loading required package: Rcpp
## Loading 'brms' package (version 2.22.0). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
##
## Attaching package: 'brms'
##
## The following object is masked from 'package:lme4':
##
##     ngrps
##
## The following object is masked from 'package:stats':
##
##     ar
```

```
library(tidyr)

#THE ACTUAL PAPER:

#structure for results ... graphs, stats, exploratory items --> i.e.
results... show the graphs, describe what the graph shows, present the
accompanying statistics and what it means, additional observations
(trail stuff eg)

#discussion ... what do my results mean for the hypothesis that I
started out with, the lit review section builds up the hypothesis I
wanna test, I come back to this now and say what I now think, what the
data from the experiment has changed about my beliefs. things id
change about the design (bigger sample, kinds of stimuli, etc), other
stuff you could do using the same setup
```

```
#conclusion... short, wheres what I thought would happen, here's what
I did, here's what I found

# +

#introduction... explain what the dissertations about, what the
quesition is, doenst have to be very long, want to get as rapidy as
possible into explaining what we already know about htis stuff. scene
setting.

#read csv
HUMAN <- read.csv("/Users/ninasmithson/Desktop/QuantMethodsSem1Yr4/
QuantMethodsSem1Yr4/data/HUM_condition.csv")
AI <- read.csv("/Users/ninasmithson/Desktop/QuantMethodsSem1Yr4/
QuantMethodsSem1Yr4/data/AI_condition.csv")

HUMAN  <- HUMAN |>
  filter(user.output!="OTH")


AI  <- AI |>
  filter(user.output!="OTH")
#why do we filter out OTH? Kenny said this is whats normally done, see
if you can find supporting evidence anywhere, posiibly in Loy and
Smith

#turning each persons response into a proportion
cat_prop <- HUMAN |>
  group_by(participant, sentance.type) |>
  summarise(
    cat_prop = sum(category) / n(),
    .groups = "drop"
  )
cat_prop

## # A tibble: 28 × 3
##    participant sentance.type cat_prop
##          <int> <chr>            <dbl>
##  1           1 DO               0
##  2           1 PO               1
##  3           2 DO               1
##  4           2 PO               0.333
##  5           3 DO               1
```

```
## 6 3 PO 0.667
## 7 4 DO 0.333
## 8 4 PO 0.6
## 9 5 DO 1
## 10 5 PO 0.6
## # ℹ 18 more rows
```

```
#turning each participants response into a proportion (but with the AI
participants)
AIcat_prop <- AI |>
  group_by(participant, sentance.type) |>
  summarise(
    AIcat_prop = sum(category) / n(),
    .groups = "drop"
  )
AIcat_prop
```

```
## # A tibble: 34 × 3
##    participant sentance.type AIcat_prop
##          <int> <chr>              <dbl>
##  1           1 DO                 1
##  2           1 PO                 0.333
##  3           2 DO                 0.6
##  4           2 PO                 0.25
##  5           3 DO                 1
##  6           3 PO                 0.333
##  7           4 DO                 0
##  8           4 PO                 1
##  9           5 DO                 0.4
## 10           5 PO                 0
## # ℹ 24 more rows
```

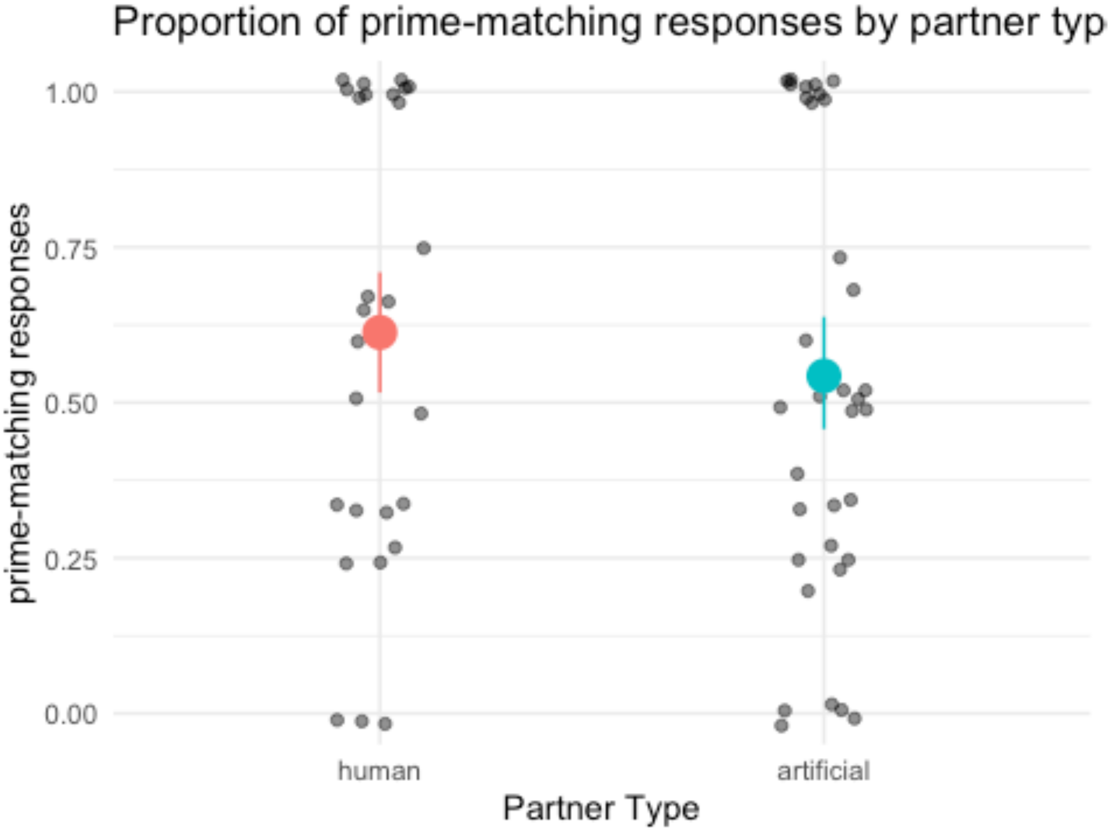
```
#this section is just transforming my combined data
#reading combined data that I redid on Sheets
NewCOMB <- read.csv("/Users/ninasmithson/Desktop/QuantMethodsSem1Yr4/
QuantMethodsSem1Yr4/data/NewCOMB.csv")

#accidentally left a space after 'artificial', so im just removing it
here
NewCOMB$partner <- trimws(NewCOMB$partner)
#getting rid of any accidental spaces before mutating the column
NewCOMB$user.output <- trimws(NewCOMB$user.output)
```

```
NewCOMB  <- NewCOMB |>
  filter(user.output!="OTH")

#turning the columns i want to work with into factors
# Convert categorical variables to factors
NewCOMB$partner <- factor(NewCOMB$partner, levels = c("human",
"artificial"))
NewCOMB$sentance.type <- factor(NewCOMB$sentance.type, levels =
c("DO", "PO"))
NewCOMB$user.output <- factor(NewCOMB$user.output, levels = c("DO",
"PO", "OTH"))

# Convert participant to a factor for random effects
NewCOMB$participant <- factor(NewCOMB$participant)

#NOTES ON THE BOXPLOT GRAPHS

#do a graph where it AI vs human and dowt worry abt DO vs PO
#just get rid of sentance type on NewCOMB data
# the lines = error bars , the bigger the less certianty
#easier to read with just proportion of DO responses, and then convert
to 'did you match your partner' for second plot.
#then bring in models

#calculating the porportional responses for each partiicpant in the
combined data
Ccat_prop <- NewCOMB |>
  group_by(partner, participant, sentance.type) |>
  summarise(
    Ccat_prop = sum(category) / n(),
    .groups = "drop"
  )
Ccat_prop

## # A tibble: 62 × 4
##    partner participant sentance.type Ccat_prop
##    <fct>   <fct>       <fct>             <dbl>
##  1 human   18          DO                0
##  2 human   18          PO                1
##  3 human   19          DO                1
##  4 human   19          PO                0.333
##  5 human   20          DO                1
##  6 human   20          PO                0.667
```

```
## 7 human 21 DO 0.333
## 8 human 21 PO 0.6
## 9 human 22 DO 1
## 10 human 22 PO 0.6
## # ℹ 52 more rows
```

```
#adding a column in my ocmbined data that takes the proportional
resposnes in relation to "DO"
NewCOMB <- NewCOMB |>
   mutate(
    DO.production = case_when(
      user.output == "DO" ~ 1,
      user.output == "PO" ~ 0,
      TRUE ~ NA_real_
    ), .before = 5
  )

#showing the combined results -- taking the 'category' responses (1
for if the partiicpant matched, and 0 if they did not) and plotting
the proportion of correct responses per partiicipant
NewCOMB |>
ggplot() +
  # Proportion of each participant
  geom_jitter(
    data = Ccat_prop,
    aes(x = partner, y = Ccat_prop),
    width = 0.1, alpha = 0.5
  ) +
  # Mean proportion by stimulus with confidence interval
  stat_summary(
    data = NewCOMB,
    aes(x = partner, y = category, colour = partner),
    fun.data = "mean_cl_boot", size = 1
  ) +
 # facet_grid(cols = vars(partner)) +

  labs(
    title = "Proportion of prime-matching responses by partner type",
    caption = "Mean proportion is represented by coloured points with
95% bootstrapped Confidence Intervals.",
    x = "Partner Type",
```

```
    y = "prime-matching responses"
  ) +
  #ylim(0, 1) +
  coord_cartesian(ylim = c(0, 1)) +
  theme(legend.position = "none")+
  theme_minimal() +
  guides(colour = "none")
```



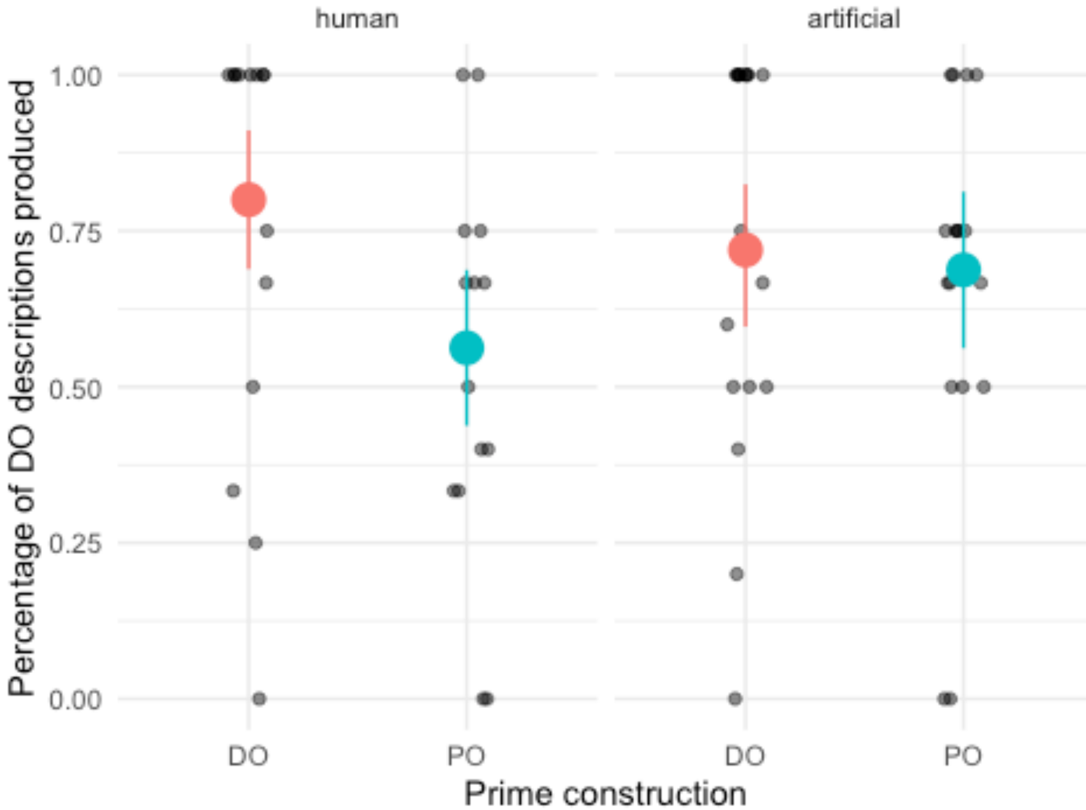
ı proportion is represented by coloured points with 95% bootstrapped Confidence Intervals.

```
#add a column that is do and its 1 if they produce the do and 0 if
they produce the PO and hten just plot the proportion of DO resposnes
and
#Y axis -- proportion of DO responses
#how often do they produce DO based off the prime
#basically i need to figure out how to add a column to my NewCOMB data
(and the other ones too probably) that rates whether the participant
responds with DO (1) or PO (0)
```

```
#what does this graph show?
#this graph takes the proportional amount of 'correct' to 'incorrect' responses ('successfully primed' and 'unsuccessfully primed' respectively) for each participant in both experimental conditions.
#The jittered points represent individual data points, and the colored circles represent the mean proportion of prime-matching responses, with confidence intervals.
#The confidence intervals around the colored circles show the range within which the true mean likely lies, with 95% confidence. The confidence intervals overlap, indicating there is a degree of uncertainty in the estimates of the mean proportions, but the human and artificial categories have different mean values. The human mean is higher, while the artificial mean is lower.

#calculating the proportional responses of DO production for each participant in the combined data
#by doing so, we can add a column where 'DO' = 1 and 'PO' = 0. Through this, we can plot the proportions of DO responses on the y-axis, and see how often participants produce DO based off the prime
# what we would expect to see is that they do it less with PO sentences, and more with DO sentences. this will be separated by partner type, so we can compare how well participants matched in the PO and DO conidtions.

Ccat_DOprop <- NewCOMB |>
  group_by(partner, participant, sentance.type) |>
  summarise(
    Ccat_DOprop = sum(DO.production) / n(),
    .groups = "drop"
  )
Ccat_DOprop
```

```
## # A tibble: 62 × 4
##    partner participant sentance.type Ccat_DOprop
##    <fct>   <fct>       <fct>               <dbl>
##  1 human   18          DO                  0
##  2 human   18          PO                  0
##  3 human   19          DO                  1
##  4 human   19          PO                  0.667
##  5 human   20          DO                  1
##  6 human   20          PO                  0.333
##  7 human   21          DO                  0.333
```

```
## 8 human  21         PO                  0.4
## 9 human  22         DO                  1
## 10 human 22         PO                  0.4
## # ℹ 52 more rows

NewCOMB |>
ggplot() +
  # Proportion of each participant
  geom_jitter(
    data = Ccat_DOprop,
    aes(x = sentance.type, y = Ccat_DOprop),
    position = position_jitter(width = 0.1, height = 0),
    alpha = 0.5
) +
  facet_grid(cols = vars(partner)) +
  # Mean proportion by stimulus with confidence interval
  stat_summary(
    data = NewCOMB,
    aes(x = sentance.type, y = DO.production, colour = sentance.type),
    fun.data = "mean_cl_boot", size = 1
  ) +

  facet_grid(cols = vars(partner)) +

  labs(
    #title = "Proportion of DO responses by prime type",
    caption = "Mean proportion is represented by coloured points with
95% bootstrapped Confidence Intervals.",
    x = "Prime construction",
    y = "Percentage of DO descriptions produced"
  ) +
  #ylim(0, 1) +
  coord_cartesian(ylim = c(0, 1)) +
  theme(legend.position = "none")+
  theme_minimal() +
  guides(colour = "none")
```



ı proportion is represented by coloured points with 95% bootstrapped Confidence Intervals.

```
#there is a higher proportion of DO responses in the human category
when the participant is DO-primed. however, the confidence intervals
overlap and are quite large. the proportion of DO responses from PO-
primed sentences is lower, which is to be expected since the hope
would be that participants produce PO responses based off PO-primes.
this being said, the teal-colored points are much higher than we would
expect.
#while we want the red points to be higher, we'd expect the teal
points to be lower. the fact that the teal point for the artificial
condition is higher indicates again a slightly weaker priming effect.
#again though, the confidence intervals are very large and overlap, so
the difference in mean points is relatively negligible. For a clearer
look into this one should run a model.

model_alt200 <- glmer(category ~ partner + (1 | prime) + (1 |
participant),
                   data = NewCOMB,
```

```r
                   family = binomial,
                   control = glmerControl(optimizer = "Nelder_Mead"))
summary(model_alt200)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: category ~ partner + (1 | prime) + (1 | participant)
##    Data: NewCOMB
## Control: glmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##    254.4    267.6   -123.2    246.4      194
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.6089 -0.7530  0.4718  0.7469  1.6593
##
## Random effects:
##  Groups      Name        Variance  Std.Dev.
##  participant (Intercept) 4.004e-08 0.0002001
##  prime       (Intercept) 8.860e-01 0.9412499
## Number of obs: 198, groups:  participant, 31; prime, 10
##
## Fixed effects:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)         0.5197     0.3822   1.360    0.174
## partnerartificial  -0.4154     0.3212  -1.293    0.196
##
## Correlation of Fixed Effects:
##             (Intr)
## partnrrtfcl -0.454
```

```r
# here, we use a Generalized Linear Mixed Effects Model, which
incorporates both fixed effects (partner) and random effects
(participant behavior/ prime), and doesn't treat the data as
individual variables.

#what do we learn from this model?
# affirms the box plot, no statistical significance in the findings.
The partherartificial fixed effect Estimate is -0.4154, which
indicates a negative relationship between artificial partner and
```

```
participant performance, however the number is quite minimal.
Additionally, Std. Error is estimated at 0.3212, which is relatively
large in comparison to the size of the estimate. This suggests that
there is substantial uncertainty in the estimate of the effect of
partnerartificial.

#Participants are less likely to give a positive response when
interacting with an artificial partner. However, the p-value of 0.196
is greater than 0.05, meaning that the effect of partnerartificial is
not statistically significant at the 5% significance level. Therefore,
the model does not provide strong evidence to reject the null
hypothesis that there is no difference in response probabilities
between the artificial and human partner conditions.

#For the artificial partner condition, the model suggests a trend
towards lower response probabilities compared to the human partner,
but this difference is not statistically significant. The Std.
indicates that any effect is very noisy, and we cant tell with a
sample of this size whether the effect is real or coincidental. One
would need further analysis or a larger sample size to more
conclusively determine if there is a significant effect.

#in conclusion, numerically, there is a bit less of a priming effect
with the AI partner, which is potentially interesting, but it is not
strongly supported in the stats. It is worth looking into it with a
larger sample size.

#The intercept (0.5197) shows a positive response when interacting
with a human partner. This means that in the human condition, the odds
of a participant matching their partner are slightly higher than the
odds of them not matching. The Correlation of Fixed Effects (-0.454)
suggests a moderate inverse relationship between an artificial partner
and priming success. This isn't an extremely strong correlation, but
it indicates some degree of negative association between the intercept
and the artificial partner effect.

ranef(model_alt200)

## $participant
##      (Intercept)
## 1   4.115370e-08
## 2  -3.775621e-08
## 3   3.902726e-08
## 4   3.086660e-08
## 5  -7.764291e-08
```

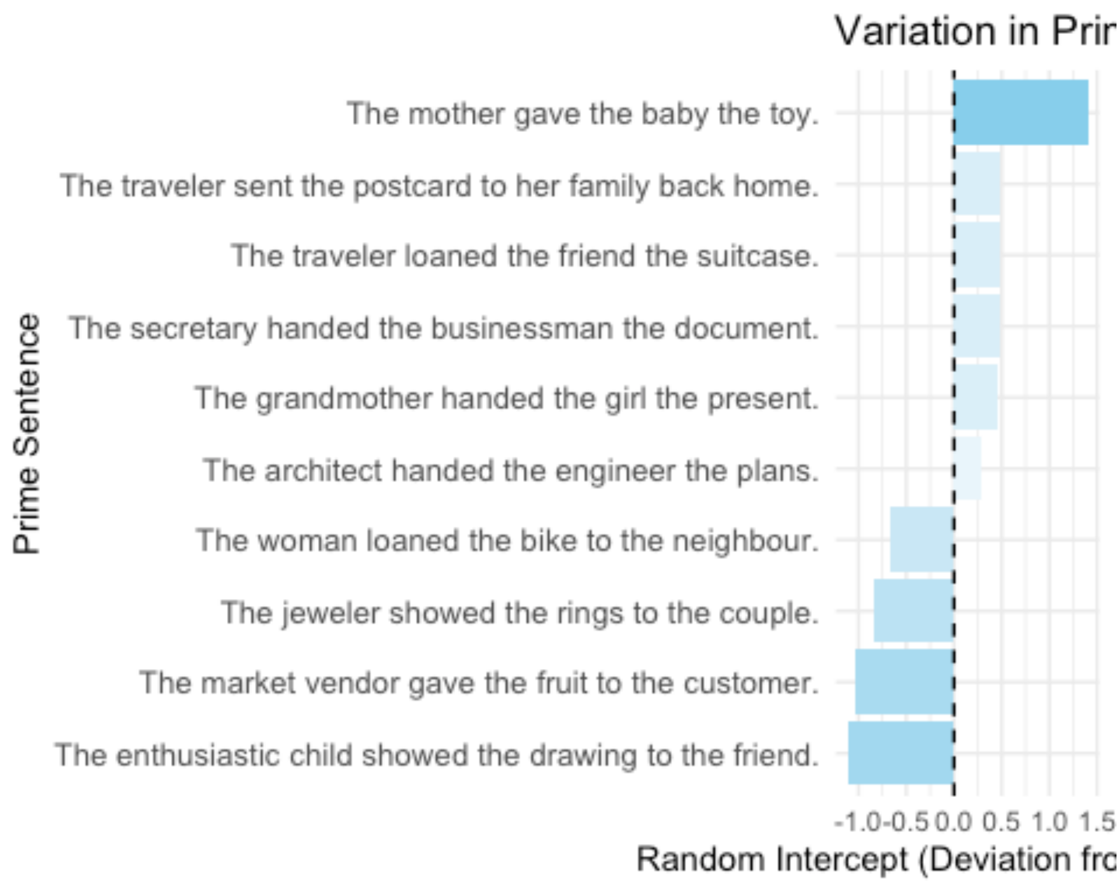
```
## 6   3.588659e-08
## 7  -1.027989e-07
## 8   4.111986e-08
## 9  -2.273840e-08
## 10 -4.267178e-08
## 11  1.115241e-09
## 12  3.903249e-08
## 13  3.165871e-08
## 14 -8.594368e-10
## 15  1.036424e-09
## 16 -2.835274e-08
## 17  5.554044e-08
## 18  1.336867e-08
## 19  8.326727e-09
## 20  5.830927e-08
## 21 -2.534256e-08
## 22  7.641341e-08
## 23 -1.270221e-08
## 24 -1.443864e-08
## 25 -3.436104e-08
## 26 -5.274066e-08
## 27 -5.917213e-08
## 28 -1.022662e-08
## 29 -2.346033e-08
## 30  8.744702e-08
## 31 -1.931163e-08
##
## $prime
##
(Intercept)
## The architect handed the engineer the plans.
0.2735370
## The enthusiastic child showed the drawing to the friend.
-1.1171424
## The grandmother handed the girl the present.
0.4635193
## The jeweler showed the rings to the couple.
-0.8379662
## The market vendor gave the fruit to the customer.
-1.0385921
## The mother gave the baby the toy.
```

```
1.3981832
## The secretary handed the businessman the document.
0.4770496
## The traveler loaned the friend the suitcase.
0.4789973
## The traveler sent the postcard to her family back home.
0.4794050
## The woman loaned the bike to the neighbour.
-0.6715578
##
## with conditional variances for "participant" "prime"
```

```
#The differences between participants is extremely small, with no
substantial participant variation (close to zero). This is likely
because the maximum data points per participant is 10, and having
removed "OTH" responses, the data points dips from 310 to 198
observations, which isn't enough to discern any meaningful variation.
#The sentence prime is a bit more interesting. (Look at different
sentences, make sure main result (priming effect of AI) is super
clear, but talk a bit about differences between the prime type
sentences and how they garner different responses if that's
interesting).

# Create a data frame with the prime sentences and their random
intercept values
prime_data <- data.frame(
  sentence = c(
    "The architect handed the engineer the plans.",
    "The enthusiastic child showed the drawing to the friend.",
    "The grandmother handed the girl the present.",
    "The jeweler showed the rings to the couple.",
    "The market vendor gave the fruit to the customer.",
    "The mother gave the baby the toy.",
    "The secretary handed the businessman the document.",
    "The traveler loaned the friend the suitcase.",
    "The traveler sent the postcard to her family back home.",
    "The woman loaned the bike to the neighbour."
  ),
  random_intercept = c(
    0.2735370, -1.1171424, 0.4635193, -0.8379662, -1.0385921,
    1.3981832, 0.4770496, 0.4789973, 0.4794050, -0.6715578
  )
)
```

```
# View the data frame
print(prime_data)

##                                                     sentence
random_intercept
## 1              The architect handed the engineer the plans.
0.2735370
## 2  The enthusiastic child showed the drawing to the friend.
-1.1171424
## 3              The grandmother handed the girl the present.
0.4635193
## 4               The jeweler showed the rings to the couple.
-0.8379662
## 5         The market vendor gave the fruit to the customer.
-1.0385921
## 6                         The mother gave the baby the toy.
1.3981832
## 7        The secretary handed the businessman the document.
0.4770496
## 8              The traveler loaned the friend the suitcase.
0.4789973
## 9   The traveler sent the postcard to her family back home.
0.4794050
## 10              The woman loaned the bike to the neighbour.
-0.6715578

prime_data$sentence <- factor(prime_data$sentence,
                              levels =
prime_data$sentence[order(prime_data$random_intercept)])

# Bar plot to show the variation in prime sentences in ascending
order, with color gradient
ggplot(prime_data, aes(x = sentence, y = random_intercept, fill =
abs(random_intercept))) +
  geom_col() +  # Create bars with a color fill based on the absolute
value of the random intercept
  coord_flip() +  # Flip the axes to make the sentences readable
  theme_minimal() +
  labs(
    title = "Variation in Prime Sentences",
    x = "Prime Sentence",
    y = "Random Intercept (Deviation from Average)"
```

```
  ) +
  theme(axis.text.y = element_text(size = 10)) +
  scale_fill_gradient2(midpoint = 0, low = "lightpink", mid = "white",
high = "skyblue") +  # Color gradient
  geom_hline(yintercept = 0, color = "black", size = 0.5, linetype =
"dashed") + #adds a line to the 0 axis
   guides(fill = "none") #gets rid of the key

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2
3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this
warning was
## generated.
```



```r
model_altD0 <- glmer(DO.production ~ partner + (1 | prime) + (1 |
participant),
                   data = NewCOMB,
                   family = binomial,
                   control = glmerControl(optimizer = "Nelder_Mead"))
summary(model_altD0)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: DO.production ~ partner + (1 | prime) + (1 | participant)
##    Data: NewCOMB
## Control: glmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##    241.7    254.8   -116.8    233.7      194
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.9634 -0.8878  0.4306  0.6033  1.2948
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.8330   0.9127
##  prime       (Intercept) 0.5057   0.7111
## Number of obs: 198, groups:  participant, 31; prime, 10
##
## Fixed effects:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.8564     0.4223   2.028   0.0426 *
## partnerartificial   0.1479     0.4807   0.308   0.7583
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## partnrrtfcl -0.607
```

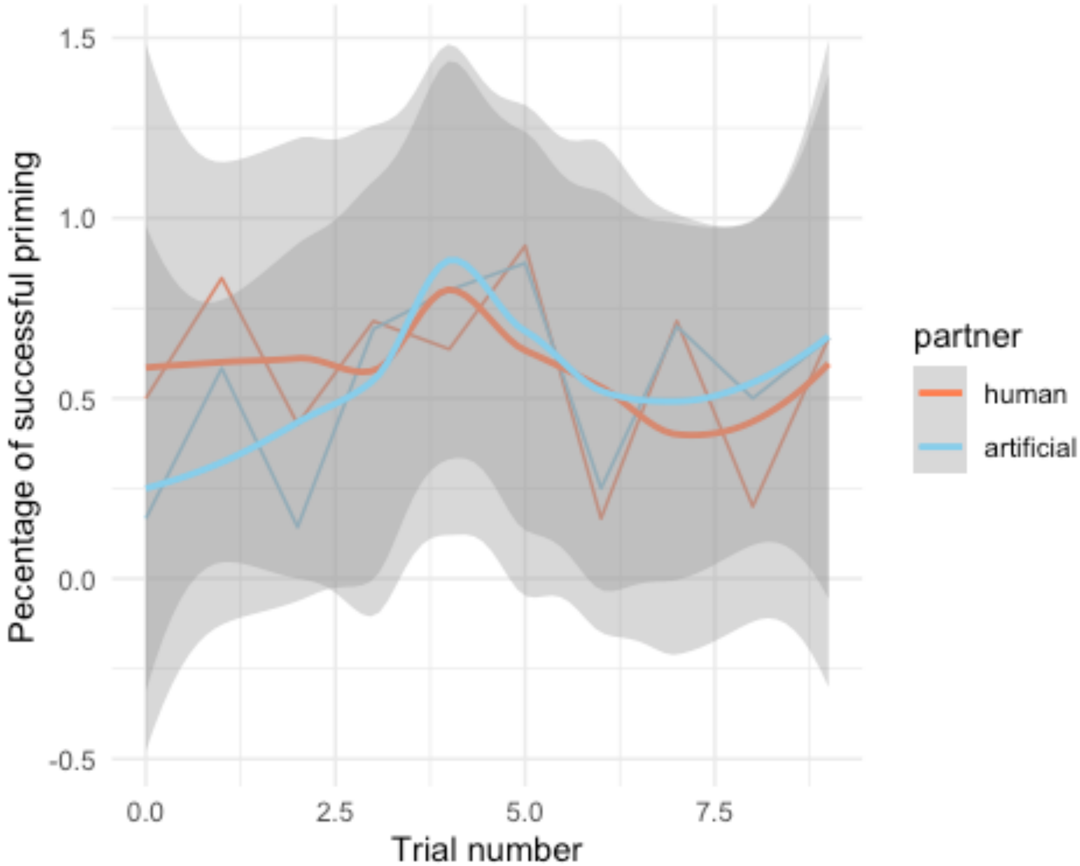
```r
# plot... x axis is trial number, y axis is category variable, if
they're constant its a. straight line, if they're matching less then
the line goes downwards.
```

```
#First we need to reformat the data so each prime sentence gets
assigned a trial number that matches the order the participants viewed
them in the experiment. This can be done using mutate() and adding a
new column
NewCOMB <- NewCOMB |>
  mutate(
    trial.no = case_when(
      prime == "The jeweler showed the rings to the couple." ~ 0,
      prime == "The secretary handed the businessman the document." ~
1,
      prime ==  "The market vendor gave the fruit to the customer." ~
2,
      prime == "The traveler loaned the friend the suitcase." ~ 3,
      prime == "The traveler sent the postcard to her family back
home. " ~ 4,
      prime == "The mother gave the baby the toy." ~ 5,
      prime == "The enthusiastic child showed the drawing to the
friend." ~ 6,
      prime == "The grandmother handed the girl the present." ~ 7,
      prime == "The woman loaned the bike to the neighbour." ~ 8,
      prime == "The architect handed the engineer the plans." ~ 9,
      TRUE ~ NA_real_
    ), .before = 5
  )

Ccat_timeprop <- NewCOMB |>
  group_by(partner, trial.no) |>
  summarise(
    Ccat_timeprop = sum(category) / n(),
    .groups = "drop"
  )
Ccat_timeprop

## # A tibble: 20 × 3
##    partner    trial.no Ccat_timeprop
##    <fct>         <dbl>         <dbl>
##  1 human             0         0.5
##  2 human             1         0.833
##  3 human             2         0.429
##  4 human             3         0.714
##  5 human             4         0.636
```

```
##  6 human              5         0.923
##  7 human              6         0.167
##  8 human              7         0.714
##  9 human              8         0.2
## 10 human              9         0.667
## 11 artificial         0         0.167
## 12 artificial         1         0.583
## 13 artificial         2         0.143
## 14 artificial         3         0.692
## 15 artificial         4         0.8
## 16 artificial         5         0.875
## 17 artificial         6         0.25
## 18 artificial         7         0.7
## 19 artificial         8         0.5
## 20 artificial         9         0.667
```

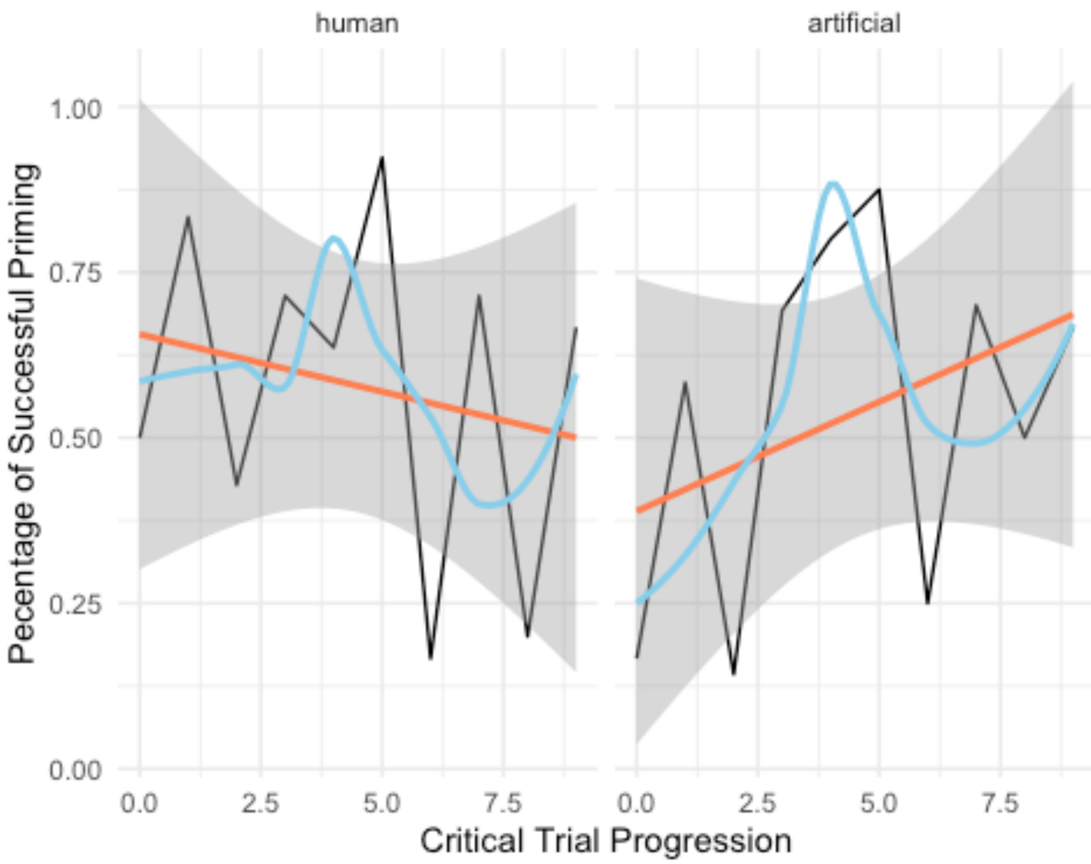
```
NewCOMB |>
  ggplot( aes(x = trial.no, y = predicted_prob, color = partner)) +
  geom_line(
    data = Ccat_timeprop,
    aes(x = trial.no, y = Ccat_timeprop, group = partner)) +
  geom_smooth(
  data = Ccat_timeprop,
  aes(x = trial.no, y = Ccat_timeprop, group = partner),
 # method = "lm",  # Use loess for a smooth, nonlinear trend line
  se = TRUE,
  #color = "hotpink"
)+

  labs(
    #title = "Variation in Prime Sentences",
    x = "Trial number",
    y = "Pecentage of successful priming"
  ) +
  #facet_grid(cols = vars(partner)) +
  theme_minimal() +
  scale_color_manual(values = c("coral", "skyblue"))
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
#This graph was done to see if there is any trend in participants
performance over time, and we can see that there is a slight downward
trend for participant accuracy in the human partner trials, and a
slight upward trend in the artificial partner trials. However, the
overlayed Local Polynomial Regression line (blue) helps visualize how
noisy the effect is, along with the large confidence interval. The
next step is to run a model to confirm the visual inspection of the
results.

NewCOMB |>
  ggplot() +
  geom_line(
    data = Ccat_timeprop,
    aes(x = trial.no, y = Ccat_timeprop)) +
  geom_smooth(
  data = Ccat_timeprop,
  aes(x = trial.no, y = Ccat_timeprop),
```

```
  method = "lm",  # Use loess for a smooth, nonlinear trend line
  se = TRUE,
  color = "coral"
)+
  geom_smooth(
  data = Ccat_timeprop,
  aes(x = trial.no, y = Ccat_timeprop),
  method = "loess",  # Use loess for a smooth, nonlinear trend line
  se = FALSE,
  color = "skyblue"
)+
  labs(
    #title = "Variation in Prime Sentences",
    x = "Critical Trial Progression",
    y = "Pecentage of Successful Priming"
  ) +
  facet_grid(cols = vars(partner)) +
  theme_minimal()

## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```




```
timemodel <- glmer(category ~ trial.no * partner + (1 | prime) + (1 |
participant),
                   data = NewCOMB,
                   family = binomial,
                   control = glmerControl(optimizer = "Nelder_Mead"))

summary(timemodel)

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: category ~ trial.no * partner + (1 | prime) + (1 |
participant)
##    Data: NewCOMB
## Control: glmerControl(optimizer = "Nelder_Mead")
##
```

```
##      AIC      BIC   logLik deviance df.resid
##    253.7    273.5   -120.9    241.7      192
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.3918 -0.7418  0.4373  0.7358  1.8950
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  participant (Intercept) 0.01189  0.1090
##  prime       (Intercept) 0.88236  0.9393
## Number of obs: 198, groups:  participant, 31; prime, 10
##
## Fixed effects:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  0.77853    0.67923   1.146   0.2517
## trial.no                    -0.07781    0.13323  -0.584   0.5592
## partnerartificial           -1.35745    0.56266  -2.413   0.0158 *
## trial.no:partnerartificial  0.25178    0.12081   2.084   0.0372 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) tril.n prtnrr
## trial.no    -0.825
## partnrrtfcl -0.411  0.355
## trl.n:prtnr  0.322 -0.435 -0.811
```

```
primemodel <- glmer(category ~ prime + (1 | partner) + (1 |
participant),
                   data = NewCOMB,
                   family = binomial,
                   control = glmerControl(optimizer = "Nelder_Mead"))
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
control$checkConv, :
## Model failed to converge with max|grad| = 0.0154731 (tol = 0.002,
component 1)
```

```
summary(primemodel)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: category ~ prime + (1 | partner) + (1 | participant)
##    Data: NewCOMB
## Control: glmerControl(optimizer = "Nelder_Mead")
##
##      AIC      BIC   logLik deviance df.resid
##    247.7    287.1   -111.8    223.7      186
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.0274 -0.6789  0.3368  0.6381  2.0039
##
## Random effects:
##  Groups      Name        Variance  Std.Dev.
##  participant (Intercept) 0.0638452 0.252676
##  partner     (Intercept) 0.0000209 0.004572
## Number of obs: 198, groups:  participant, 31; partner, 2
##
## Fixed effects:
##
Estimate
## (Intercept)
0.7070
## primeThe enthusiastic child showed the drawing to the friend.
-2.0270
## primeThe grandmother handed the girl the present.
0.1885
## primeThe jeweler showed the rings to the couple.
-1.4060
## primeThe market vendor gave the fruit to the customer.
-1.6358
## primeThe mother gave the baby the toy.
1.4783
## primeThe secretary handed the businessman the document.
0.1944
## primeThe traveler loaned the friend the suitcase.
0.1637
## primeThe traveler sent the postcard to her family back home.
0.2131
## primeThe woman loaned the bike to the neighbour.
```

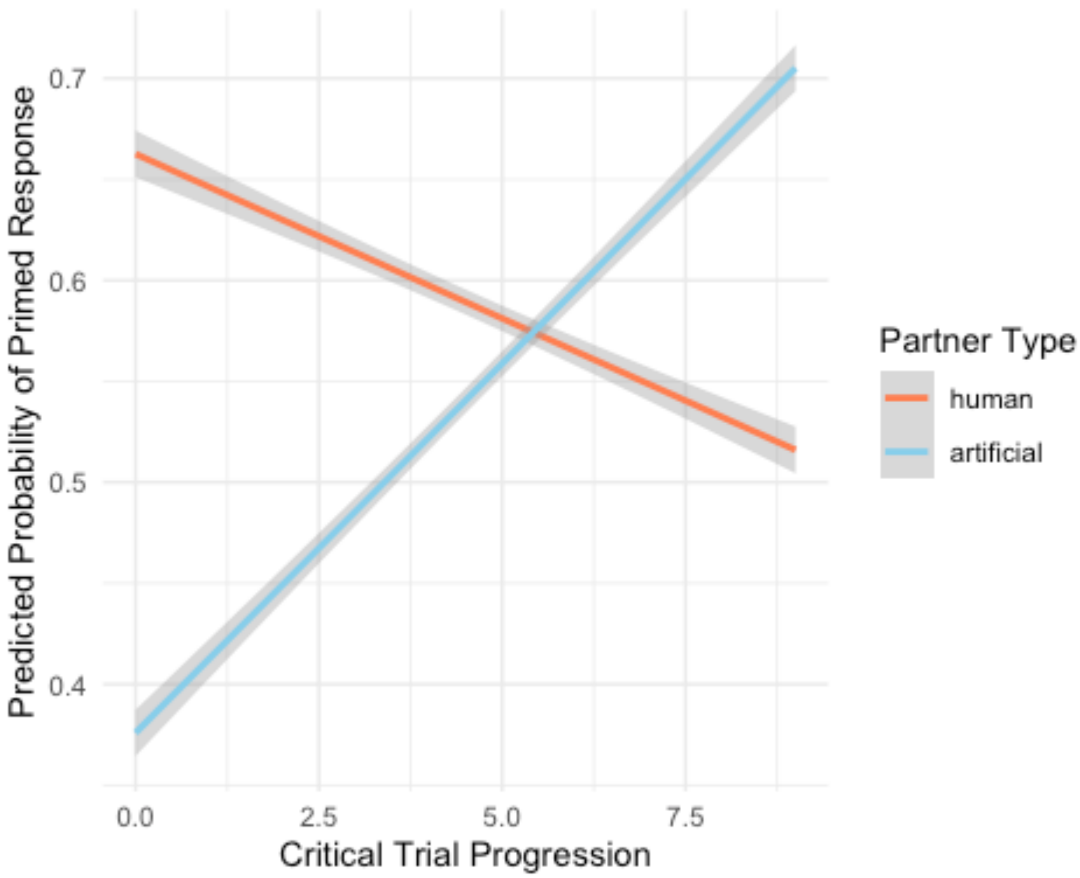
```
-1.4029
##                                                               Std.
Error
## (Intercept)
0.6231
## primeThe enthusiastic child showed the drawing to the friend.
0.9099
## primeThe grandmother handed the girl the present.
0.8196
## primeThe jeweler showed the rings to the couple.
0.7612
## primeThe market vendor gave the fruit to the customer.
0.7542
## primeThe mother gave the baby the toy.
0.8709
## primeThe secretary handed the businessman the document.
0.7669
## primeThe traveler loaned the friend the suitcase.
0.7914
## primeThe traveler sent the postcard to her family back home.
0.7889
## primeThe woman loaned the bike to the neighbour.
0.9475
##                                                               z
value Pr(>|z|)
## (Intercept)
1.135   0.2565
## primeThe enthusiastic child showed the drawing to the friend.
-2.228   0.0259
## primeThe grandmother handed the girl the present.
0.230   0.8181
## primeThe jeweler showed the rings to the couple.
-1.847   0.0647
## primeThe market vendor gave the fruit to the customer.
-2.169   0.0301
## primeThe mother gave the baby the toy.
1.697   0.0896
## primeThe secretary handed the businessman the document.
0.254   0.7999
## primeThe traveler loaned the friend the suitcase.
0.207   0.8362
## primeThe traveler sent the postcard to her family back home.
0.270   0.7871
```

```
## primeThe woman loaned the bike to the neighbour.
-1.481   0.1387
##
## (Intercept)
## primeThe enthusiastic child showed the drawing to the friend. *
## primeThe grandmother handed the girl the present.
## primeThe jeweler showed the rings to the couple.              .
## primeThe market vendor gave the fruit to the customer.        *
## primeThe mother gave the baby the toy.                        .
## primeThe secretary handed the businessman the document.
## primeThe traveler loaned the friend the suitcase.
## primeThe traveler sent the postcard to her family back home.
## primeThe woman loaned the bike to the neighbour.
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) pecstdttf pghtgtp pjstrttc pmvgtfttc pmgtbtt
pshtbtd ptltfts
## pTecstdttf. -0.686
## prmTghtgtp. -0.749  0.512
## prTjstrttc. -0.816  0.566     0.613
## pTmvgtfttc. -0.826  0.573     0.618   0.679
## prmTmgtbtt. -0.703  0.477     0.537   0.574    0.578
## prmTshtbtd. -0.803  0.549     0.609   0.657    0.663     0.574
## prmTtltfts. -0.775  0.528     0.591   0.633    0.638     0.558
0.631
## pTtstpthfbh -0.784  0.540     0.592   0.643    0.650     0.556
0.634   0.613
## prTwltbttn. -0.654  0.452     0.493   0.537    0.543     0.461
0.527   0.508
##             ptstpthfbh
## pTecstdttf.
## prmTghtgtp.
## prTjstrttc.
## pTmvgtfttc.
## prmTmgtbtt.
## prmTshtbtd.
## prmTtltfts.
## pTtstpthfbh
## prTwltbttn.  0.515
```

```
## optimizer (Nelder_Mead) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 0.0154731 (tol = 0.002,
component 1)
```

```
# Generate a new data frame with all possible combinations of
trial.no, partner, and prime
new_data <- expand.grid(
  trial.no = seq(0, 9, by = 1),  # Assuming trials are numbered from 1
to 10
  partner = c("human", "artificial"),
  prime = unique(NewCOMB$prime),  # Use unique primes from the
original dataset
  participant = unique(NewCOMB$participant)
)

# Predict the probabilities using the model
new_data$predicted_prob <- predict(timemodel, newdata = new_data, type
= "response")

#this is the code i went with for this dumb graph
ggplot(new_data, aes(x = trial.no, y = predicted_prob, color =
partner)) +
 # stat_summary(fun = mean, geom = "line", size = 1.0, aes(group =
partner)) +  # Mean line for each partner
  geom_smooth(
    data = new_data,
    aes(x = trial.no, y = predicted_prob, group = partner),
    method = "lm",
    se = TRUE  # Confidence interval (shaded area)
  ) +
  labs(
    x = "Critical Trial Progression",
    y = "Predicted Probability of Primed Response",
    color = "Partner Type"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("coral", "skyblue"))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Predicted Probability of Primed Response
0.7
0.6
0.5
0.4
0.0
2.5
5.0
7.5
Critical Trial Progression
Partner Type
human
artificial