Physical Pragmatics (decider_3)

Preprocessing

```r
# Read in the participant data.
data_0 = read_csv(file.path(data_path, "raw_data.csv"), quote="~")

# Convert the JSON string into JSON.
data_1 = lapply(data_0$data, fromJSON)

# Extract the trial information for each participant and stack them.
data_3 = tibble()
for (p in 1:length(data_1)) {
  # Trim the map and add the participant ID back in.
  data_2 = data_1[p][[1]]$trials %>%
    as.data.frame() %>%
    gather(type, num, trial_num, exclusion_num) %>%
    na.omit() %>%
    mutate(type=gsub("num", "", type)) %>%
    mutate(target=as.numeric(target), trial=paste(type, num, sep=""),
           object=data_1[p][[1]]$setup$object,
           condition=data_1[p][[1]]$setup$condition,
           unique_id=data_1[p][[1]]$id,
           age=as.numeric(data_1[p][[1]]$subject_information$age)) %>%
    select(-type, -num)

  # Stack the trial information for the current participant.
  data_3 = rbind(data_3, data_2)
}

# Exclude participants that said the unmodified door was harder to walk
# through.
data_5 = tibble()
exclusions = tibble()
for (participant in unique(data_3$unique_id)) {
  data_4 = data_3 %>%
    filter(unique_id==participant)

  if (filter(data_4, trial=="exclusion_1")[,"target"]==0) {
    exclusions = rbind(exclusions, data_4)
  }
  else {
    data_5 = rbind(data_5, data_4)
  }
}

# Write the preprocessed data.
```

```
write_csv(data_5, file.path(data_path, "data.csv"))
```

Compute Door Endorsement

a priori Labels

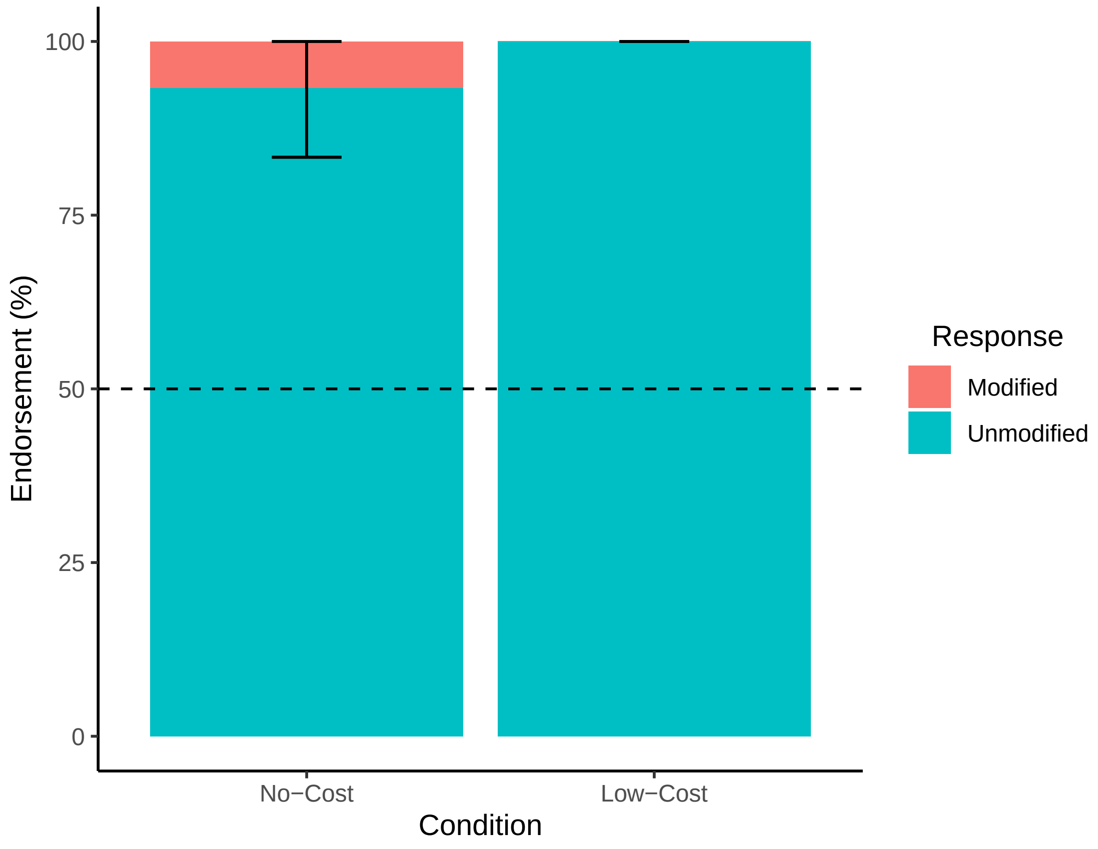
First, we'll compute the participant endorsement (N=60, M=35.72 years, SD=12.48 years) for the unmodified door (i.e., that the modified door should be avoided) using our *a priori* labels of door difficulty.

```
# Read in the preprocessed data.
data_6 = read_csv(file.path(data_path, "data.csv"))

# Filter the trial of interest.
data_7 = data_6 %>%
  filter(trial=="trial_1")

# Set up the bootstrap functions.
compute_mean = function(data, indices) {
  return(mean(data[indices]))
}

compute_bootstrap = function(data) {
  simulations = boot(data=data$target,
                     statistic=compute_mean,
                     R=10000)

  return(boot.ci(simulations, type="perc")$perc)
}

# Set the seed and compute the 95% bootstrapped CIs.
set.seed(seed)
ci = data.frame()
bootstrap_data = compute_bootstrap(filter(data_7, condition=="none"))
ci = rbind(ci, data.frame(lower_ci=bootstrap_data[4]*100,
                          upper_ci=bootstrap_data[5]*100,
                          condition="none"))

bootstrap_data = compute_bootstrap(filter(data_7, condition=="low"))
```

```
## [1] "All values of t are equal to  1 \n Cannot calculate confidence intervals"
```

```
ci = rbind(ci, data.frame(lower_ci=ifelse(!is.null(bootstrap_data),
                                          bootstrap_data[4]*100, 100),
                          upper_ci=ifelse(!is.null(bootstrap_data),
                                          bootstrap_data[5]*100, 100),
                          condition="low"))

# Convert individual judgments to aggregate percentages.
data_8 = data_7 %>%
  group_by(condition) %>%
  summarize(unmodified=sum(target)/n()*100, modified=100-unmodified) %>%
  gather(response, endorsement, unmodified, modified) %>%
  left_join(ci)
```

```
# Plot the data (using our labels of door difficulty).
plot_0 = data_8 %>%
  ggplot(aes(x=condition, y=endorsement, fill=response)) +
  geom_bar(stat="identity") +
  geom_errorbar(aes(ymin=lower_ci, ymax=upper_ci), width=0.2) +
  geom_hline(yintercept=50, linetype="dashed", color="black") +
  theme_classic() +
  theme(aspect.ratio=1.0,
        legend.title=element_text(hjust=0.5)) +
  ylab("Endorsement (%)") +
  scale_x_discrete(name="Condition",
                   limits=c("none", "low"),
                   labels=c("No-Cost", "Low-Cost")) +
  scale_fill_discrete(name="Response",
                      limits=c("modified", "unmodified"),
                      labels=c("Modified", "Unmodified"))
plot_0
```



```
# Use a Fisher's exact test to compute probability that participants would
# avoid the modified door by chance.
data_9 = data_7 %>%
  mutate(target=ifelse(target==0, "modified", "unmodified")) %>%
  group_by(condition, target) %>%
  summarize(total=n()) %>%
  rbind(data.frame(condition="low", target="modified", total=0)) %>%
  arrange(condition, target)
```

```
fisher.test(rbind(filter(data_9, target=="unmodified")$total,
                  filter(data_9, target=="modified")$total))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  rbind(filter(data_9, target == "unmodified")$total, filter(data_9, target == "modified")$tota
## p-value = 0.4915
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.188835      Inf
## sample estimates:
## odds ratio
##        Inf
```

```
# Compute a binomial test with the alternative hypothesis that the participant
# endorsement of the unmodified door is above chance.
# NOTE: Computing a two-sided binomial test for "conservativeness".
data_10 = filter(data_7, condition=="none")
binom.test(x=sum(data_10$target), n=length(data_10$target), p=0.5,
           alternative="two.sided")
```

```
##
##  Exact binomial test
##
## data:  sum(data_10$target) and length(data_10$target)
## number of successes = 28, number of trials = 30, p-value = 8.68e-07
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7792646 0.9918219
## sample estimates:
## probability of success
##              0.9333333
```

```
data_11 = filter(data_7, condition=="low")
binom.test(x=sum(data_11$target), n=length(data_11$target), p=0.5,
           alternative="two.sided")
```

```
##
##  Exact binomial test
##
## data:  sum(data_11$target) and length(data_11$target)
## number of successes = 30, number of trials = 30, p-value = 1.863e-09
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.8842967 1.0000000
## sample estimates:
## probability of success
##                      1
```

```
# Compute a t-test with the alternative hypothesis that the participant
# endorsement of the unmodified door is above chance.
t.test(x=data_10$target, n=length(data_10$target), mu=0.5,
       alternative="greater")
```

```
##
```

```
##  One Sample t-test
##
## data:  data_10$target
## t = 9.3551, df = 29, p-value = 1.464e-10
## alternative hypothesis: true mean is greater than 0.5
## 95 percent confidence interval:
##  0.8546288       Inf
## sample estimates:
## mean of x
## 0.9333333
```

```
# Compute a t-test with the alternative hypothesis that the participant
# endorsement of the unmodified door in the low-cost condition is higher than
# in the no-cost condition.
t.test(x=data_11$target, y=data_10$target, alternative="greater")
```

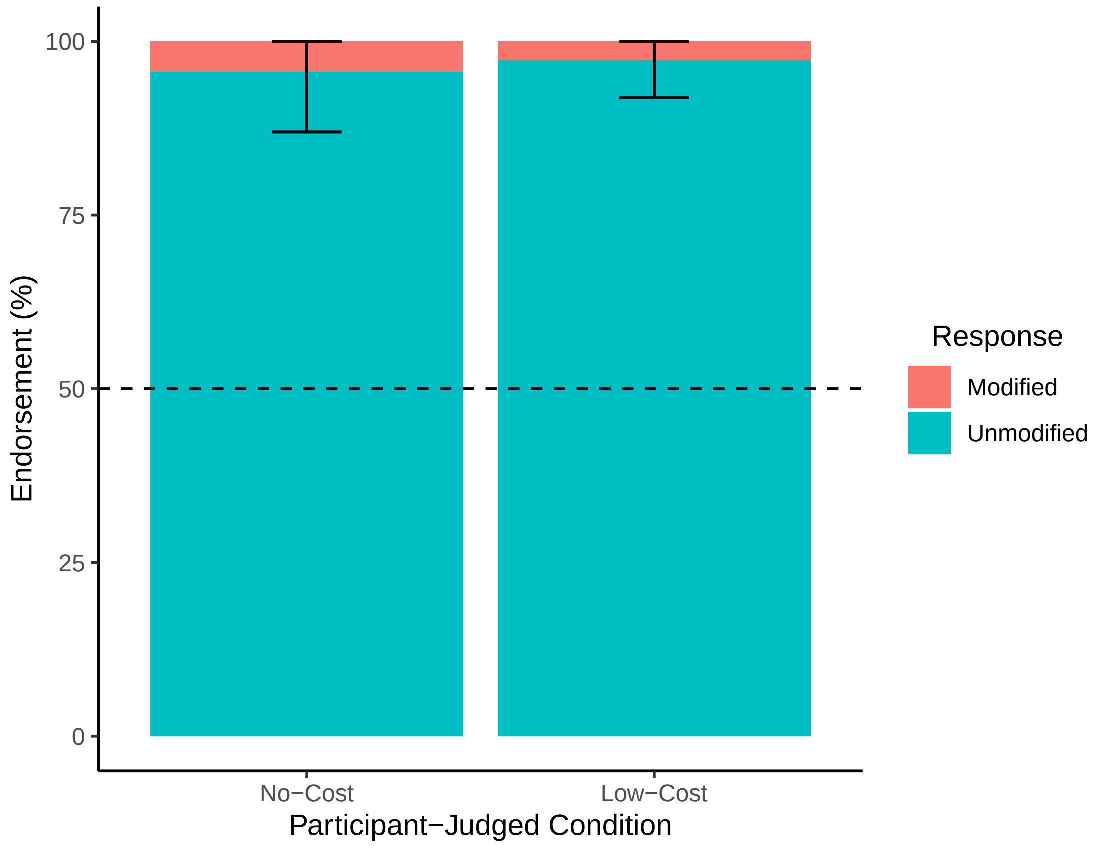
```
##
##  Welch Two Sample t-test
##
## data:  data_11$target and data_10$target
## t = 1.4392, df = 29, p-value = 0.08039
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.01203784         Inf
## sample estimates:
## mean of x mean of y
## 1.0000000 0.9333333
```

Participant Labels

Now, we'll analyze the participant endorsement (N=60, M=35.72 years, SD=12.48 years) for the unmodified door (i.e., that the modified door should be avoided) using participant labels of door difficulty.

```
# Filter the trial of interest.
data_12 = data_6 %>%
  spread(trial, target) %>%
  mutate(condition=ifelse(exclusion_1==1, "low", "none")) %>%
  select(-exclusion_1, -exclusion_2) %>%
  rename(target=trial_1)

# Set the seed and compute the 95% bootstrapped CIs.
set.seed(seed)
ci = data.frame()
bootstrap_data = compute_bootstrap(filter(data_12, condition=="none"))
ci = rbind(ci, data.frame(lower_ci=bootstrap_data[4]*100,
                          upper_ci=bootstrap_data[5]*100,
                          condition="none"))

bootstrap_data = compute_bootstrap(filter(data_12, condition=="low"))
ci = rbind(ci, data.frame(lower_ci=ifelse(!is.null(bootstrap_data),
                                          bootstrap_data[4]*100, 100),
                          upper_ci=ifelse(!is.null(bootstrap_data),
                                          bootstrap_data[5]*100, 100),
                          condition="low"))
```

```r
# Convert individual judgments to aggregate percentages.
data_13 = data_12 %>%
  group_by(condition) %>%
  summarize(unmodified=sum(target)/n()*100, modified=100-unmodified) %>%
  gather(response, endorsement, unmodified, modified) %>%
  left_join(ci)

# Plot the data (using participant labels of door difficulty).
plot_1 = data_13 %>%
  ggplot(aes(x=condition, y=endorsement, fill=response)) +
  geom_bar(stat="identity") +
  geom_errorbar(aes(ymin=lower_ci, ymax=upper_ci), width=0.2) +
  geom_hline(yintercept=50, linetype="dashed", color="black") +
  theme_classic() +
  theme(aspect.ratio=1.0,
        legend.title=element_text(hjust=0.5)) +
  ylab("Endorsement (%)") +
  scale_x_discrete(name="Participant-Judged Condition",
                   limits=c("none", "low"),
                   labels=c("No-Cost", "Low-Cost")) +
  scale_fill_discrete(name="Response",
                      limits=c("modified", "unmodified"),
                      labels=c("Modified", "Unmodified"))
plot_1
```



```r
# Use a Fisher's exact test to compute probability that participants would
# avoid the modified door by chance.
data_14 = data_12 %>%
  mutate(target=ifelse(target==0, "modified", "unmodified")) %>%
  group_by(condition, target) %>%
  summarize(total=n()) %>%
  rbind(data.frame(condition="low", target="modified", total=0)) %>%
  arrange(condition, target)
fisher.test(rbind(filter(data_14, target=="unmodified")$total,
                  filter(data_14, target=="modified")$total))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  rbind(filter(data_14, target == "unmodified")$total, filter(data_14, target == "modified")$to
## p-value = 1
## alternative hypothesis: two.sided
```

```r
# Compute a binomial test with the alternative hypothesis that the participant
# endorsement of the unmodified door is above chance.
# NOTE: Computing a two-sided binomial test for "conservativeness".
data_15 = filter(data_12, condition=="none")
binom.test(x=sum(data_15$target), n=length(data_15$target), p=0.5,
           alternative="two.sided")
```

```
##
##  Exact binomial test
##
## data:  sum(data_15$target) and length(data_15$target)
## number of successes = 22, number of trials = 23, p-value = 5.722e-06
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7805134 0.9988998
## sample estimates:
## probability of success
##              0.9565217
```

```r
data_16 = filter(data_12, condition=="low")
binom.test(x=sum(data_16$target), n=length(data_16$target), p=0.5,
           alternative="two.sided")
```

```
##
##  Exact binomial test
##
## data:  sum(data_16$target) and length(data_16$target)
## number of successes = 36, number of trials = 37, p-value = 5.53e-10
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.8583969 0.9993160
## sample estimates:
## probability of success
##               0.972973
```

```r
# Compute a t-test with the alternative hypothesis that the participant
# endorsement of the unmodified door is above chance.
t.test(x=data_15$target, n=length(data_15$target), mu=0.5,
       alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  data_15$target
## t = 10.5, df = 22, p-value = 2.466e-10
## alternative hypothesis: true mean is greater than 0.5
## 95 percent confidence interval:
##  0.8818633       Inf
## sample estimates:
## mean of x
## 0.9565217
```

```
# Compute a t-test with the alternative hypothesis that the participant
# endorsement of the unmodified door in the low-cost condition is higher than
# in the no-cost condition.
t.test(x=data_16$target, y=data_15$target, alternative="greater")
```

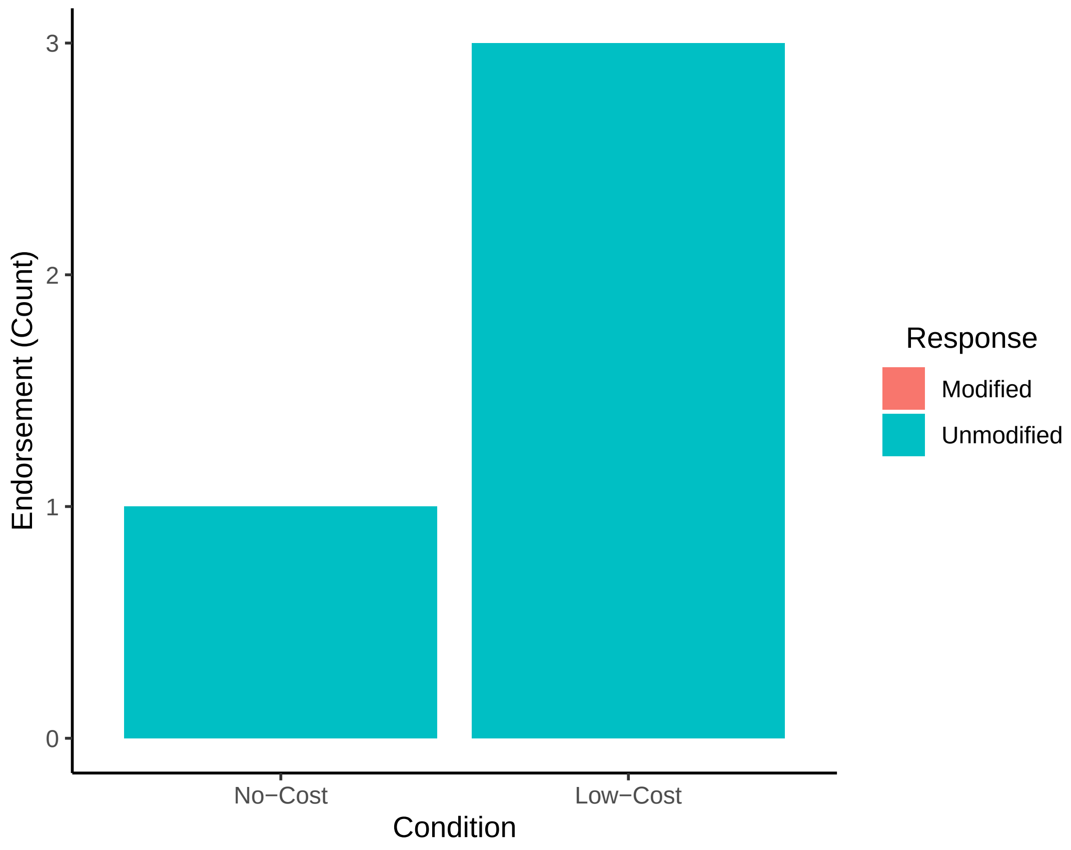
```
##
##  Welch Two Sample t-test
##
## data:  data_16$target and data_15$target
## t = 0.32135, df = 38.751, p-value = 0.3748
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.06981764         Inf
## sample estimates:
## mean of x mean of y
## 0.9729730 0.9565217
```

Analyze Excluded Participants

a priori Labels

Here we analyze the data from excluded participants, using *a priori* labels of door difficulty, to see if their judgments align with our main results.

```
# Extract the participants who said the unmodified door was more difficult to
# walk through.
data_17 = exclusions %>%
  filter(trial=="trial_1")

# Sum individual judgments by condition and door type.
data_18 = data_17 %>%
  mutate(target=ifelse(target==0, "modified", "unmodified")) %>%
  group_by(condition, target) %>%
  summarize(count=n())

# Plot the data (using our labels of door difficulty).
plot_2 = data_18 %>%
  ggplot(aes(x=condition, y=count, fill=target)) +
  geom_histogram(stat="identity", position="dodge") +
  theme_classic() +
  theme(aspect.ratio=1.0,
        legend.title=element_text(hjust=0.5)) +
  scale_x_discrete(name="Condition",
                   limits=c("none", "low"),
                   labels=c("No-Cost", "Low-Cost")) +
  ylab("Endorsement (Count)") +
  scale_fill_discrete(name="Response",
                      limits=c("modified", "unmodified"),
                      labels=c("Modified", "Unmodified"))
plot_2
```


Endorsement (Count)
No−Cost
Low−Cost
Condition
Response
Modified
Unmodified