

Physical Pragmatics (decider_0)

Preprocessing

Preprocess both partitions of participant data (a second experiment was needed to round out the pre-registered sample size).

```
# Read in the first partition of the participant data.
data_0 = read_csv(file.path(data_0_path, "raw_data.csv"))

# Update old formatting and remove irrelevant columns.
data_1 = data_0 %>%
  rename(condition=CostCondition, object=Object, side=Counterbalancing,
          costlier=HighCost, possible=Difficulty, response=DoorChoice,
          unique_id=UserID) %>%
  select(unique_id, condition, object, side, costlier, possible, response)
data_1$side[which(data_1$side=="0")] = "right"
data_1$side[which(data_1$side=="1")] = "left"
data_1$costlier[which(data_1$costlier=="BaseDoor")] = "unmodified"
data_1$costlier[which(data_1$costlier=="TargetDoor")] = "modified"
data_1$possible[which(data_1$possible=="Yes")] = 1
data_1$possible[which(data_1$possible=="No")] = 0
data_1$response[which(data_1$response=="ClearDoor")] = 1
data_1$response[which(data_1$response=="ModifiedDoor")] = 0

# Remove missing entries.
data_2 = data_1 %>%
  filter(condition!="") %>%
  na.omit()

# Add participant numbers to make it easier to distinguish participants across
# the disjoint datasets.
data_3 = data_2 %>%
  rownames_to_column() %>%
  mutate(rowname=as.integer(as.integer(rowname)-1),
         possible=as.integer(possible),
         response=as.integer(response)) %>%
  rename(participant=rowname) %>%
  select(-unique_id) %>%
  mutate(age=NA)

# Write the preprocessed data for the first partition.
write_csv(data_3, file.path(data_0_path, "data.csv"))

# Read in the preprocessed data for the first partition.
data_3 = read_csv(file.path(data_0_path, "data.csv"))

# Read in the second partition of the participant data.
data_4 = read_csv(file.path(data_1_path, "raw_data.csv")) %>%
```

```

mutate(target=as.integer(substr(target, 2, 2)))

# Read in the participant age information.
data_5 = read_csv(file.path(data_1_path, "subject_information.csv")) %>%
  select(workerid, age) %>%
  mutate(age=as.numeric(gsub("\\", "", age)))

# Combine the trial and exclusion columns.
data_6 = data_4 %>%
  gather(trial_type, num, trial_num, exclusion_num) %>%
  na.omit() %>%
  mutate(trial=gsub("num", "", paste(trial_type, num, sep=""))) %>%
  select(-trial_type, -num) %>%
  arrange(workerid) %>%
  left_join(data_5)

# Import the setup information to know which side the low-cost door was on.
setup_0 = read_tsv(file.path(data_1_path, "trial_information.tsv")) %>%
  select(workerid, Answer.setup)

# Remove the quotes, backslashes, and braces.
setup_1 = setup_0 %>%
  mutate(Answer.setup=gsub("\\", "", Answer.setup, fixed=TRUE)) %>%
  mutate(Answer.setup=gsub("\\", "", Answer.setup, fixed=TRUE)) %>%
  mutate(Answer.setup=gsub("{", "", Answer.setup, fixed=TRUE)) %>%
  mutate(Answer.setup=gsub("}", "", Answer.setup, fixed=TRUE))

# Extract the side information.
setup_2 = setup_1 %>%
  separate(Answer.setup, into=c(NA, "side", NA, NA), sep=",") %>%
  mutate(side=gsub("side:", "", side))

# Append the side information.
data_7 = data_6 %>%
  spread(trial, target) %>%
  rename(costlier=exclusion_1, possible=exclusion_2, response=trial_1) %>%
  left_join(setup_2) %>%
  rename(participant=workerid) %>%
  mutate(participant=participant+nrow(data_3))

# Rename the factors for readability.
data_7$costlier[which(data_7$costlier==0)] = "unmodified"
data_7$costlier[which(data_7$costlier==1)] = "modified"
data_7$costlier[which(data_7$costlier==2)] = "equal"

# Write the preprocessed data for the second partition.
write_csv(data_7, file.path(data_1_path, "data.csv"))

# Combine the two data partitions.
data_8 = data_3 %>%
  rbind(data_7)

# Exclude participants who said the unmodified door was more difficult to walk

```

```
# through.
data_9 = data_8 %>%
  filter(costlier!="unmodified")

# Chop off the extra participant in the low-cost condition.
data_10 = data_9 %>%
  filter(participant!=161)
```

Compute Door Endorsement

a priori Labels

First, we'll compute the participant endorsement ($N=160$, $M=34.85$ years, $SD=8.38$ years) for the unmodified door (i.e., that the modified door should be avoided) using our *a priori* labels of door difficulty.

```
# Compute the (percent) participant endorsement of each door in each condition.
data_11 = data_10 %>%
  do(left_join(., summarize(group_by(., condition),
                                unmodified=sum(response==1),
                                modified=sum(response==0),
                                total=n())) %>%
  mutate(unmodified=unmodified/total*100,
         modified=modified/total*100)

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

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

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

# Compute the bootstrapped 95% CIs for participant endorsement of the
# unmodified door for both conditions.
set.seed(seed)
ci = data.frame()
bootstrap_data = compute_bootstrap(filter(data_11, condition=="none")$response)
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_11, condition=="low")$response)
ci = rbind(ci, data.frame(lower_ci=bootstrap_data[4]*100,
                        upper_ci=bootstrap_data[5]*100,
                        condition="low"))

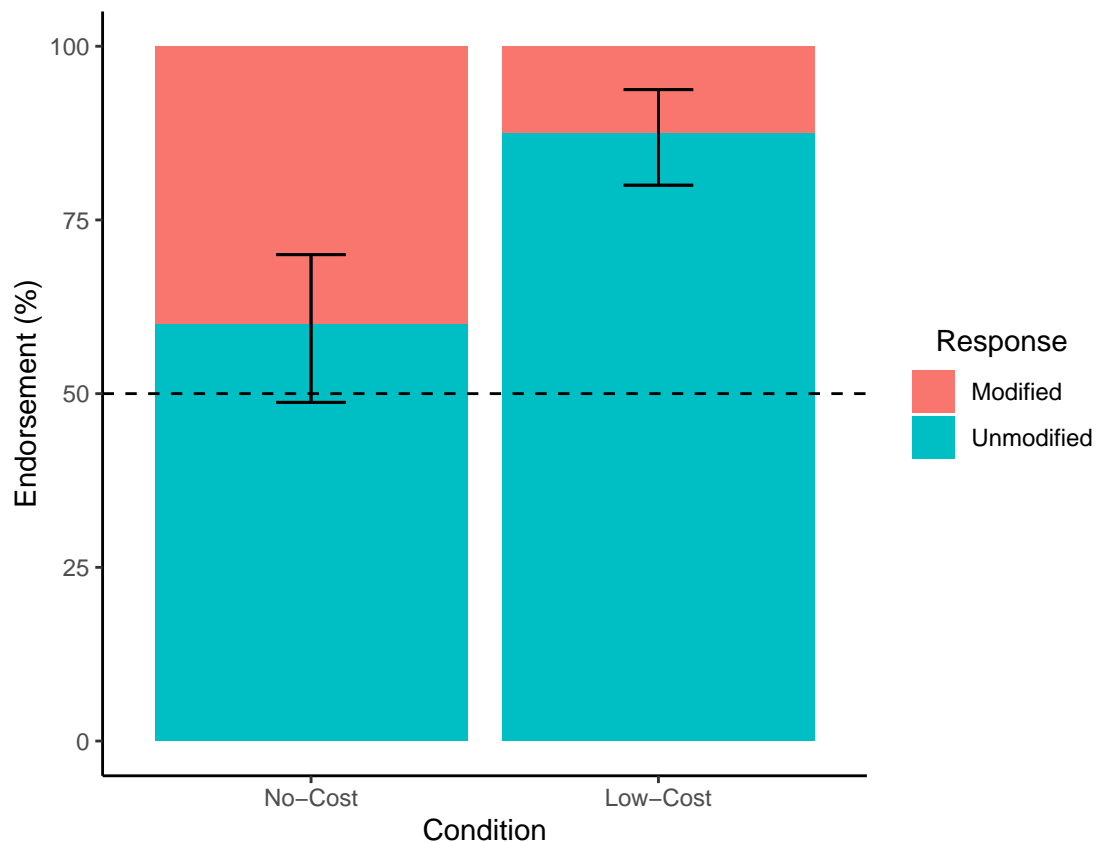
# Compute the percentage participant endorsement for each door per condition.
data_12 = data_11 %>%
```

```

gather(response, endorsement, unmodified, modified) %>%
group_by(condition, endorsement, response) %>%
summarize(total=n()) %>%
left_join(ci) %>%
select(-total)

# Plot the data (using our labels of door difficulty).
plot_0 = data_12 %>%
  ggplot(aes(x=condition, y=endorsement, fill=response)) +
  geom_histogram(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)) +
  scale_x_discrete(name="Condition",
                  limits=c("none", "low"),
                  labels=c("No-Cost", "Low-Cost")) +
  ylab("Endorsement (%)") +
  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_13 = data_10 %>%
  mutate(response=ifelse(response==0, "modified", "unmodified")) %>%
  group_by(condition, response) %>%
  summarize(total=n())
fisher.test(rbind(filter(data_13, response=="unmodified")$total,
                     filter(data_13, response=="modified")$total))

##
## Fisher's Exact Test for Count Data
##
## data:  rbind(filter(data_13, response == "unmodified")$total, filter(data_13, response == "modified")$total)
## p-value = 0.0001255
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  1.989338 11.578410
## sample estimates:
## odds ratio
##  4.621041

# Compute a mixed-effects logistic regression with objects as random
# intercepts.
mem_0 = glmer(response~condition+(1|object), data=data_11, family="binomial")
summary(mem_0)

## Generalized linear mixed model fit by maximum likelihood (Laplace
## Approximation) [glmerMod]
## Family: binomial ( logit )
## Formula: response ~ condition + (1 | object)
## Data: data_11
##
##           AIC          BIC    logLik deviance df.resid
##      167.8       177.1     -80.9    161.8      157
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.5617 -0.7917  0.2808  0.5756  1.2631
##
## Random effects:
## Groups Name             Variance Std.Dev.
## object (Intercept) 0.5838    0.7641
## Number of obs: 160, groups: object, 8
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.1544    0.4565   4.719 2.37e-06 ***
## conditionnone -1.6949    0.4332  -3.913 9.12e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr)
## conditionnn -0.665

# Compute a binomial test for each condition.
# NOTE: Computing a two-sided binomial test for "conservativeness".

```

```

data_14 = filter(data_11, condition=="none")
binom.test(x=sum(data_14$response), n=length(data_14$response), p=0.5,
           alternative="two.sided")

##
## Exact binomial test
##
## data: sum(data_14$response) and length(data_14$response)
## number of successes = 48, number of trials = 80, p-value = 0.09291
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.4843769 0.7079906
## sample estimates:
## probability of success
##                0.6

data_15 = filter(data_11, condition=="low")
binom.test(x=sum(data_15$response), n=length(data_15$response), p=0.5,
           alternative="two.sided")

##
## Exact binomial test
##
## data: sum(data_15$response) and length(data_15$response)
## number of successes = 70, number of trials = 80, p-value = 3.161e-12
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7821109 0.9383979
## sample estimates:
## probability of success
##                0.875

# Compute the bootstrapped 95% CIs for the participant endorsement of whether
# it would be possible to walk through the modified door for each condition.
set.seed(seed)
bootstrap_data = compute_bootstrap(filter(data_11, condition=="none")$possible)
lower_ci = bootstrap_data[4]*100
upper_ci = bootstrap_data[5]*100
possible = sum(filter(data_11, condition=="none")$possible)
not_possible = length(filter(data_11, condition=="none")$possible) - possible
possible_endorsement = possible / (possible+not_possible) * 100
print(c(possible_endorsement, lower_ci, upper_ci))

## [1] 83.75 75.00 91.25

bootstrap_data = compute_bootstrap(filter(data_11, condition=="low")$possible)
lower_ci = bootstrap_data[4]*100
upper_ci = bootstrap_data[5]*100
possible = sum(filter(data_11, condition=="low")$possible)
not_possible = length(filter(data_11, condition=="low")$possible) - possible
possible_endorsement = possible / (possible+not_possible) * 100
print(c(possible_endorsement, lower_ci, upper_ci))

## [1] 87.50 80.00 93.75

```

Participant Labels

Now, we'll compute the participant endorsement ($N=160$, $M=34.85$ years, $SD=8.38$ years) for the unmodified door (i.e., that the modified door should be avoided) using their labels of door difficulty.

```
# Compute the (percent) participant endorsement of each door in each condition.
data_16 = data_10 %>%
  do(left_join(., summarize(group_by(., costlier),
                                unmodified=sum(response==1),
                                modified=sum(response==0),
                                total=n())) %>%
    mutate(unmodified=unmodified/total*100,
           modified=modified/total*100))

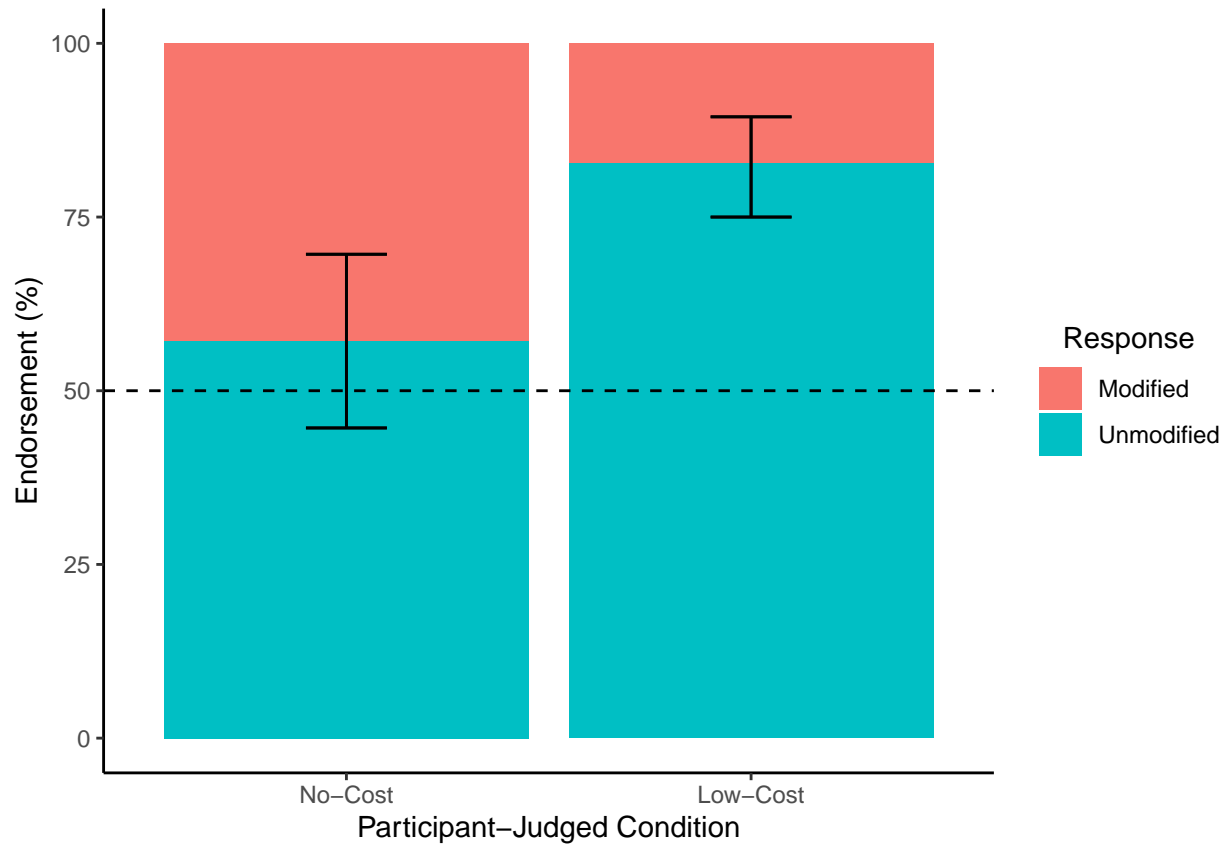
# Compute the bootstrapped 95% CI for participant endorsement of the
# unmodified door with participant labels of costliness.
set.seed(seed)
ci = data.frame()
bootstrap_data = compute_bootstrap(filter(data_16,
                                          costlier=="equal")$response)
ci = rbind(ci, data.frame(lower_ci=bootstrap_data[4]*100,
                          upper_ci=bootstrap_data[5]*100,
                          condition="equal"))

bootstrap_data = compute_bootstrap(filter(data_16,
                                          costlier=="modified")$response)
ci = rbind(ci, data.frame(lower_ci=bootstrap_data[4]*100,
                          upper_ci=bootstrap_data[5]*100,
                          condition="modified"))

# Compute the percentage participant endorsement for each door per condition.
data_17 = data_16 %>%
  gather(response, endorsement, unmodified, modified) %>%
  group_by(costlier, endorsement, response) %>%
  summarize(total=n()) %>%
  ungroup() %>%
  rename(condition=costlier) %>%
  select(-total) %>%
  left_join(ci)

# Plot the data (using participant labels of door difficulty).
plot_1 = data_17 %>%
  ggplot(aes(x=condition, y=endorsement, fill=response)) +
  geom_histogram(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(legend.title=element_text(hjust=0.5)) +
  scale_x_discrete(name="Participant-Judged Condition",
                  limits=c("equal", "modified"),
                  labels=c("No-Cost", "Low-Cost")) +
  ylab("Endorsement (%)") +
  scale_fill_discrete(name="Response",
                     limits=c("modified", "unmodified"),
                     labels=c("Modified", "Unmodified"))
```

plot_1



```
# Compute a mixed-effects logistic regression with objects as random
# intercepts.
mem_1 = glmer(response~costlier+(1|object), data=data_16, family="binomial")
summary(mem_1)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
## Approximation) [glmerMod]
## Family: binomial ( logit )
## Formula: response ~ costlier + (1 | object)
## Data: data_16
##
##      AIC      BIC    logLik deviance df.resid
##    172.4    181.6    -83.2   166.4      157
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -3.3578 -0.7470  0.3444  0.6042  1.3387
##
## Random effects:
##  Groups Name   Variance Std.Dev.
##  object (Intercept) 0.5538  0.7442
## Number of obs: 160, groups:  object, 8
##
## Fixed effects:
```



```
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.3101      0.3911   0.793  0.42796
## costliermodified  1.4330      0.4055   3.534  0.00041 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr)
## costlirmdfd -0.521

# Compute a binomial test for each condition.
# NOTE: Computing a two-sided binomial test for "conservativeness".
data_18 = filter(data_16, costlier=="equal")
binom.test(x=sum(data_18$response), n=length(data_18$response), p=0.5,
           alternative="two.sided")

##
## Exact binomial test
##
## data: sum(data_18$response) and length(data_18$response)
## number of successes = 32, number of trials = 56, p-value = 0.3497
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.4321590 0.7028791
## sample estimates:
## probability of success
##           0.5714286

data_19 = filter(data_16, costlier=="modified")
binom.test(x=sum(data_19$response), n=length(data_19$response), p=0.5,
           alternative="two.sided")

##
## Exact binomial test
##
## data: sum(data_19$response) and length(data_19$response)
## number of successes = 86, number of trials = 104, p-value = 8.22e-12
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.7403265 0.8940758
## sample estimates:
## probability of success
##           0.8269231
```

Object-based Analysis

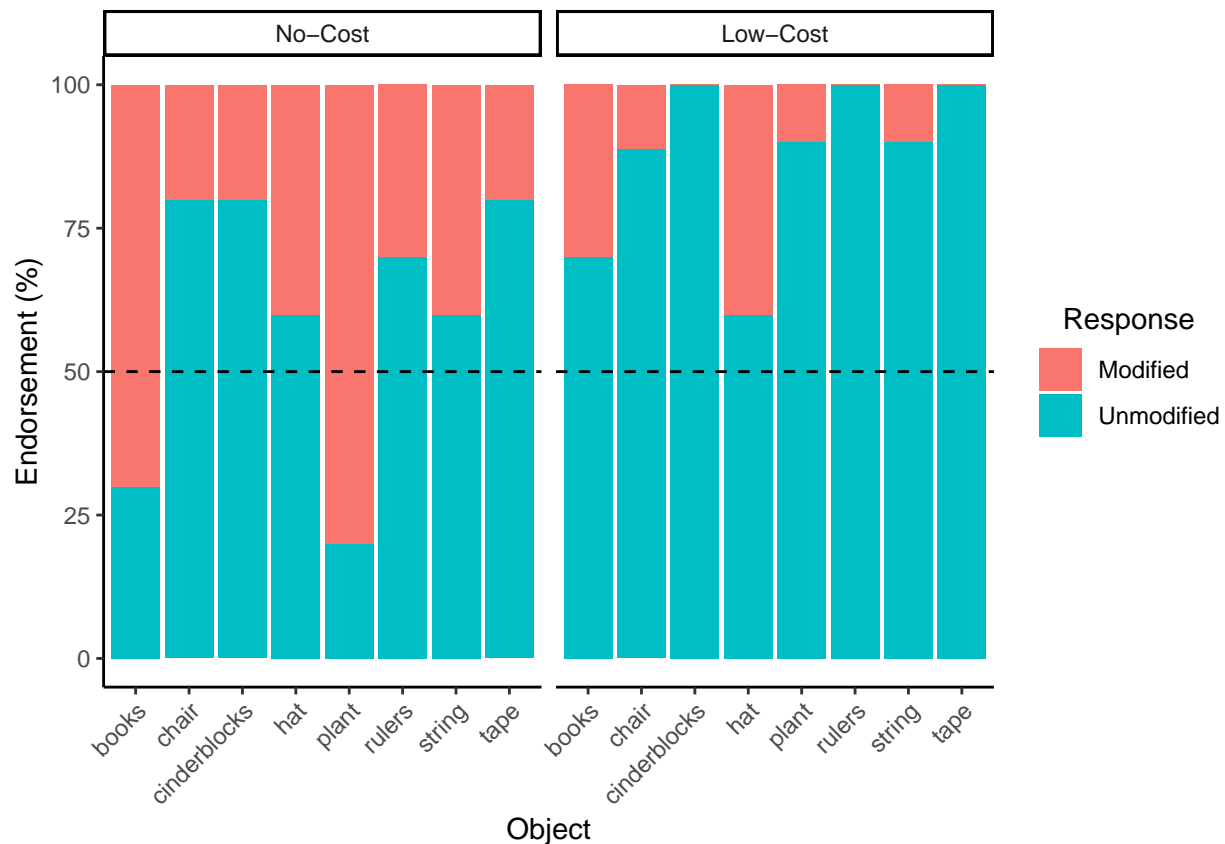
Next, we'll compute the participant endorsement ($N=160$, $M=34.85$ years, $SD=8.38$ years) of the unmodified door per object type, starting with our labels of door difficulty.

```
# For additional analysis, compute the endorsement across objects in the first
# trial.
data_20 = data_11 %>%
  group_by(condition, object) %>%
  summarize(unmodified=sum(response)/n()*100, modified=100-unmodified) %>%
  gather(door, endorsement, unmodified, modified)
```

```
# Plot the data per object (using participant labels of door difficulty).
```

```
plot_2 = data_20 %>%
  ggplot(aes(x=object, y=endorsement, fill=door)) +
  geom_histogram(stat="identity") +
  geom_hline(yintercept=50, linetype="dashed", color="black") +
  theme_classic() +
  theme(axis.text.x=element_text(angle=45, hjust=1),
        legend.title=element_text(hjust=0.5)) +
  facet_wrap(~factor(condition,
                     levels=c("none", "low"),
                     labels=c("No-Cost", "Low-Cost")))) +
  xlab("Object") +
  ylab("Endorsement (%)") +
  scale_fill_discrete(name="Response",
                     limits=c("modified", "unmodified"),
                     labels=c("Modified", "Unmodified"))
```

```
plot_2
```



Lastly, we'll compute the participant endorsement ($N=160$, $M=34.85$ years, $SD=8.38$ years) of the unmodified door per object type, using their labels of door difficulty.

```
# For additional analysis, compute the endorsement across objects in the first
# trial.
```

```
data_21 = data_16 %>%
  group_by(costlier, object) %>%
  summarize(unmodified=sum(response)/n()*100, modified=100-unmodified) %>%
```

```
gather(door, endorsement, unmodified, modified)

# Plot the data per object (using participant labels of door difficulty).
plot_3 = data_21 %>%
  ggplot(aes(x=object, y=endorsement, fill=door)) +
  geom_histogram(stat="identity") +
  geom_hline(yintercept=50, linetype="dashed", color="black") +
  theme_classic() +
  theme(axis.text.x=element_text(angle=45, hjust=1),
        legend.title=element_text(hjust=0.5)) +
  facet_wrap(~factor(costlier,
                    levels=c("equal", "modified"),
                    labels=c("No-Cost", "Low-Cost"))) +
  xlab("Object") +
  ylab("Endorsement (%)") +
  scale_fill_discrete(name="Response",
                    limits=c("modified", "unmodified"),
                    labels=c("Modified", "Unmodified"))

plot_3
```

