Image Inference - Experiment 2

Preprocessing

```r
# Read in the participant data (after manually removing errors).
data_0 = read_csv(file.path(human_path, "data.csv"), quote="~")

# Read in the MTurk results file.
mturk_results = read_csv(file.path(human_path, "mturk_results.csv"), col_names=TRUE) %>%
  mutate(Answer.surveycode=substr(Answer.surveycode, 6, length(Answer.surveycode))) %>%
  filter(AssignmentStatus=="Approved")

# Check for invalid survey codes.
invalid = sum(!(mturk_results$Answer.surveycode %in% data_0$unique_id))
if (invalid != 0) { stop("There's an invalid survey code.") }

# Extract the trial information for each participant and save it.
age = c()
for (p in 1:length(mturk_results$Answer.surveycode)) {
  # Find the database entry that contains the experiment code that this
  # participant submitted.
  data_1 = data_0 %>%
    filter(unique_id==mturk_results$Answer.surveycode[p]) %>%
    filter(nchar(results)==min(nchar(results)))

  # Extract and store this participant's age.
  age = c(age, as.integer(fromJSON(data_1$results)$subject_information$age))

  # Extract the experiment code that this participant submitted.
  experiment_code = substr(fromJSON(data_1$results)$catch_trials$experiment_code, 6, 13)

  # Find the database entry that contains the trial information for this
  # participant.
  data_2 = data_0 %>%
    filter(unique_id==experiment_code) %>%
    filter(nchar(results)==max(nchar(results)))

  # Convert the JSON string to a data frame and refactor.
  data_3 = fromJSON(data_2$results)$trials %>%
    select(-trial_num) %>%
    mutate(participant=as.character(p-1)) %>%
    select(participant, map, coords)

  # Save the trial information for this participant.
  write_csv(data_3, file.path(human_path, paste(p-1, ".csv", sep="")))
}
```

Model Comparison

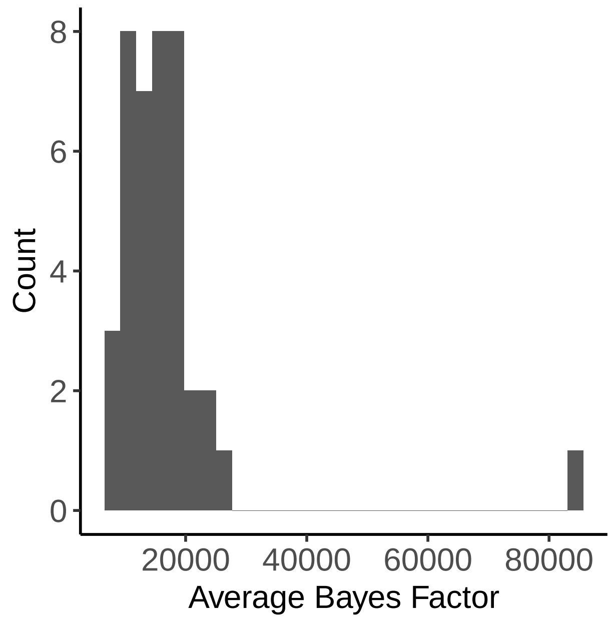
Here we pass in participant-generated paths ($M = 38.25$ years, $SD = 11.02$ years) to our model and an alternative model and compute the posterior probability of each model generating each participant-generated path. We then divide the posterior probability from our model by that of the alternative model to compute the Bayes factor. A Bayes factor greater than one indicates our model explains participant judgments better; a Bayes factor less than one indicates the alternative model explains participant judgments better.

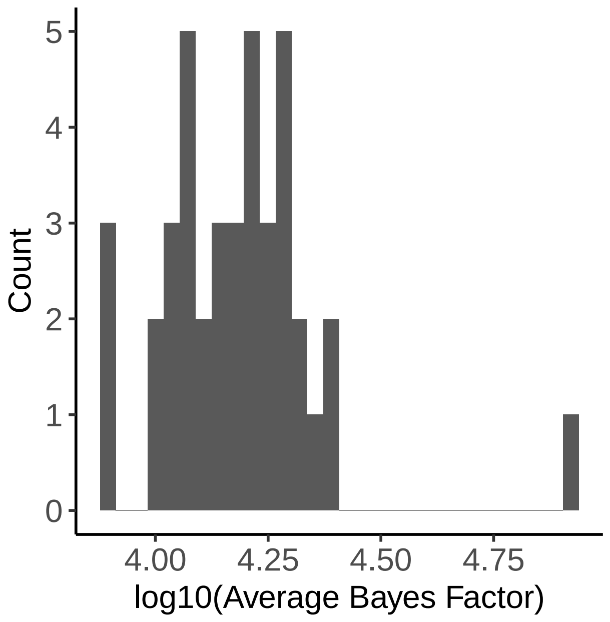
With a Bayes factor for each path for every participant, we then average them so that each participant has a single mean Bayes factor. First, we run a t-test comparing these mean Bayes factors against 0 to validate (from an NHST perspective) whether or not our model outperforms the alternative model.

```
##
##  One Sample t-test
##
## data:  data_5$mean_bayes_factor
## t = 9.1024, df = 39, p-value = 1.712e-11
## alternative hypothesis: true mean is greater than 1
## 95 percent confidence interval:
##  13800.74      Inf
## sample estimates:
## mean of x
##  16935.33
```

Next, we take a look at the distribution of mean Bayes factors among our 40 participants, both using a standard scale and in log scale.

```
plot_0 = data_5 %>%
  ggplot(aes(x=mean_bayes_factor)) +
  geom_histogram() +
  theme_classic() +
  theme(aspect.ratio=1.0,
        axis.text=element_text(size=12),
        axis.title=element_text(size=12)) +
  xlab("Average Bayes Factor") +
  ylab("Count")

plot_1 = data_5 %>%
  mutate(log_mean_bayes_factor=log10(mean_bayes_factor)) %>%
  ggplot(aes(x=log_mean_bayes_factor)) +
  geom_histogram() +
  theme_classic() +
  theme(aspect.ratio=1.0,
        axis.text=element_text(size=12),
        axis.title=element_text(size=12)) +
  xlab("log10(Average Bayes Factor)") +
  ylab("Count")

plot_2 = plot_grid(plot_0, plot_1, nrow=1)
plot_2
```





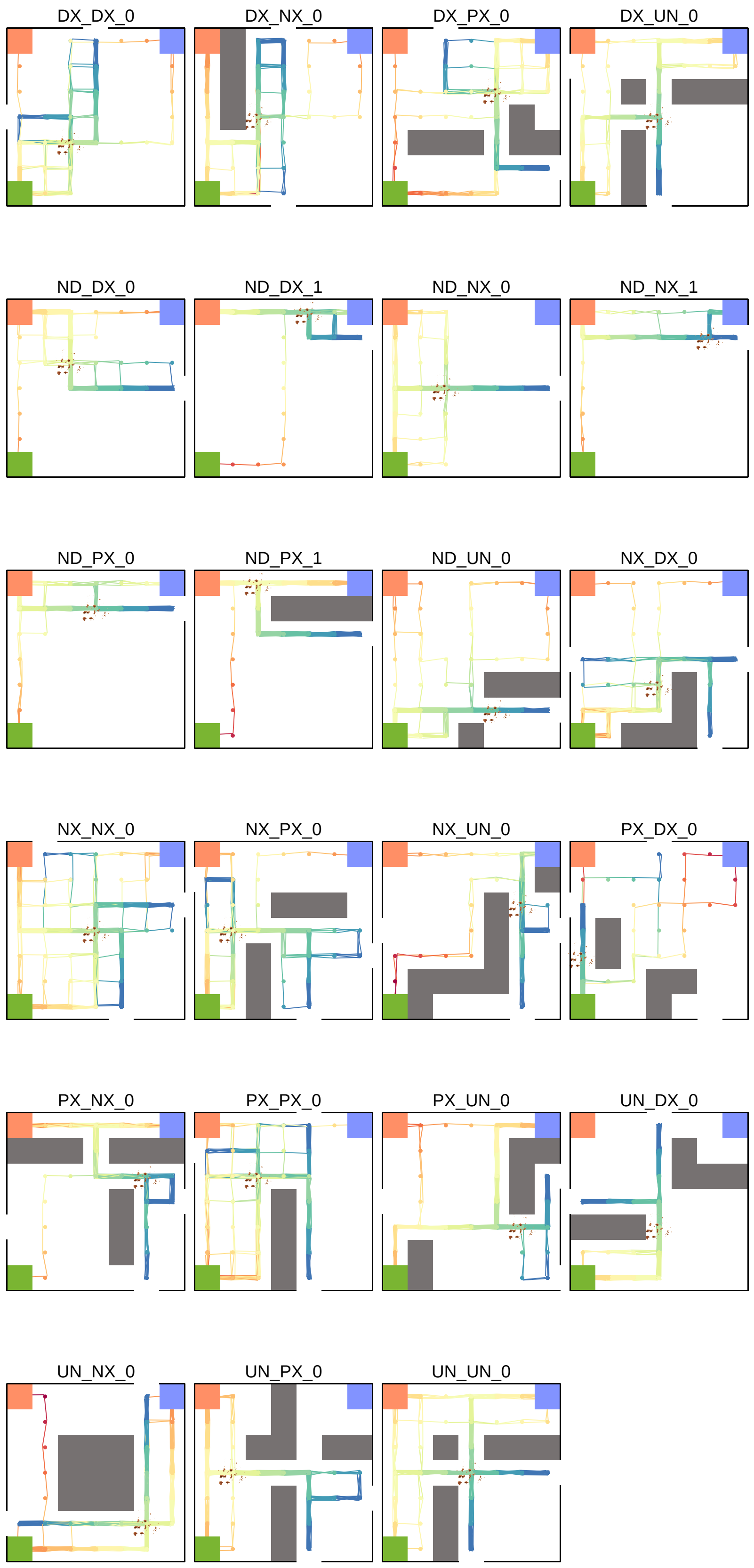
Visualizations

Here we plot the visualizations of the participant-generated data on each trial.

```
# Import the participant data.
data_6 = tibble()
for (i in 1:40) {
  # Stitch together the path to this participant's data.
  data_path = paste("data/experiment_2/human/data_1/", i-1, ".csv", sep="")

  # Import this participant's data.
  data_6 = data_6 %>%
    rbind(read_csv(data_path, col_types="dcc"))
}

# Rearrange the data such that the maps are ordered alphabetically within each
# participant.
data_7 = data_6 %>%
  arrange(participant, map)

# Restructure the participant data.
data_10 = tibble()
for (p in 1:length(unique(data_7$participant))) {
  for (m in 1:length(unique(data_7$map))) {
    # Select a participant and map.
```

```r
    data_8 = data_7 %>%
      filter(participant==unique(data_7$participant)[p],
             map==unique(data_7$map)[m])

    # Extract the x- and y-coordinates from their corresponding indices.
    indices_x = seq(1, nchar(data_8$coords), 4)
    indices_y = seq(3, nchar(data_8$coords), 4)
    x = c()
    y = c()
    for (i in 1:length(indices_x)) {
      x = c(x, as.integer(substr(data_8$coords, indices_x[i], indices_x[i])))
      y = c(y, as.integer(substr(data_8$coords, indices_y[i], indices_y[i])))
    }

    # Set up the new structure of the participant data.
    data_9 = data.frame(
      participant = unique(data_7$participant)[p],
      map = unique(data_7$map)[m],
      x = x,
      y = y,
      time = 1:length(x)
    )

    # Stack this participant's data with the rest.
    data_10 = data_10 %>%
      rbind(data_9)
  }
}

# Import the coordinates of the observations, inner walls, and outer walls.
source("stimuli/experiment_2/map_specifications.R")

# Define the coordinates of the goals.
goals = data.frame(
  x_1 = c(2.5, 8.5, 2.5),
  x_2 = c(3.5, 9.5, 3.5),
  y_1 = c(2.5, 2.5, 8.5),
  y_2 = c(3.5, 3.5, 9.5),
  label = c("A", "B", "C")
)

# Define the coordinate bounds.
map_height = 11
map_width = 11

# Set up the color palette and then the gradient we're going to visualize with.
palette = colorRampPalette(rev(brewer.pal(11, "Spectral")))
color_gradient = scale_color_gradientn(colours=palette(100), limits=c(0, 15))

# Generate the visualizations of the participant-generated paths for each
# trial.
plot_4 = list()
plot_5 = list()
```

```r
for (trial_num in 1:23) {
  # Extract the coordinates of the observation for this trial.
  crumbs = data.frame(
    x = observations[[trial_num]][1],
    y = observations[[trial_num]][2],
    filename = "stimuli/experiment_2/crumbs.png"
  )

  # Generate the visualization for this trial.
  plot_3 = data_10 %>%
    filter(map==maps[trial_num]) %>%
    mutate(x=x+1, y=y+1) %>%
    ggplot() +
    geom_path(aes(x=x, y=y, group=participant, color=time),
              position=position_jitter(height=0.1, width=0.1)) +
    geom_point(aes(x=x, y=y, color=time)) +
    scale_x_continuous(name="", limits=c(0, map_width+1)) +
    scale_y_reverse(name="", limits=c(map_height+1, 1)) +
    theme_void() +
    theme(legend.position="none",
          axis.title=element_blank(),
          axis.text=element_blank(),
          axis.ticks=element_blank()) +
    coord_fixed() +
    geom_image(data=crumbs, aes(x=x, y=y, image=filename), size=0.1) +
    geom_rect(data=goals,
              aes(xmin=x_1, xmax=x_2, ymin=y_1, ymax=y_2, fill=label),
              alpha=1.0) +
    color_gradient +
    scale_fill_manual(values=c("#FF8F66", "#8293FF", "#7AB532", "#767171"))

  # Add the walls to the visualization.
  for (wall in walls[trial_num][[1]]) {
    plot_3 = plot_3 + wall
  }

  # Add the border to the visualization.
  for (segment in segments[trial_num][[1]]) {
    plot_3 = plot_3 + segment
  }

  # Store the plot separately before adding a title and adjusting the margins.
  plot_4[[trial_num]] = plot_3

  # Add a title to the visualization, adjust the margins, and store it.
  plot_5[[trial_num]] = plot_3 +
    ggtitle(maps[trial_num]) +
    theme(plot.title=element_text(size=20, hjust=0.5, vjust=-20.0),
          plot.margin=unit(c(-2, -3, -2, -3), "cm"))
}

# Plot the visualizations in a grid.
plot_6 = plot_grid(plot_5[[1]], plot_5[[2]], plot_5[[3]], plot_5[[4]],
                   plot_5[[5]], plot_5[[6]], plot_5[[7]], plot_5[[8]],
                   plot_5[[9]], plot_5[[10]], plot_5[[11]], plot_5[[12]],
                   plot_5[[13]], plot_5[[14]], plot_5[[15]], plot_5[[16]],
                   plot_5[[17]], plot_5[[18]], plot_5[[19]], plot_5[[20]],
                   plot_5[[21]], plot_5[[22]], plot_5[[23]],
                   ncol=4)
plot_6
```


DX_DX_0
DX_NX_0
DX_PX_0
DX_UN_0
ND_DX_0
ND_DX_1
ND_NX_0
ND_NX_1
ND_PX_0
ND_PX_1
ND_UN_0
NX_DX_0
NX_NX_0
NX_PX_0
NX_UN_0
PX_DX_0
PX_NX_0
PX_PX_0
PX_UN_0
UN_DX_0
UN_NX_0
UN_PX_0
UN_UN_0

Saving the visualizations

Here we save the visualizations (excluding the title and margin adjustments).

```r
# Save the visualizations.
for (trial_num in 1:23) {
  # Stitch the path for this visualization.
  data_path = paste("analysis/experiment_2/visualizations/",
                    maps[trial_num], ".pdf", sep="")

  # Save this visualization.
  ggsave(data_path, plot_4[[trial_num]], device="pdf", height=4.5, width=7.3)
}
```