Pilot Analysis: Building a Probit Regression Model of Decisions

In this document, we build a probit regression model of intervention decisions. This model is basically a cumulative Gaussian psychometric function which estimates two parameters: 1. The mean or the *point of subjective equality*, at which users see the new machine as having equal utility to the old machine. Differences between this parameter and the utility optimal decision rule for the intervention problem reflect bias in decision-making, either toward or away from intervening. 2. The standard deviation or the amount of *noise in the perception of utility*, which reflects the consistency with which users evaluate prospects.

Load and Prepare Data

We load worker responses from our pilot and do some preprocessing.

```
# read in data
full_df <- read_csv("pilot-anonymous.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   workerId = col_character(),
##   batch = col_integer(),
##   condition = col_character(),
##   start_gain_frame = col_character(),
##   numeracy = col_integer(),
##   gender = col_character(),
##   age = col_character(),
##   education = col_character(),
##   chart_use = col_character(),
##   intervene = col_integer(),
##   outcome = col_character(),
##   pSup = col_integer(),
##   trial = col_character(),
##   trialIdx = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

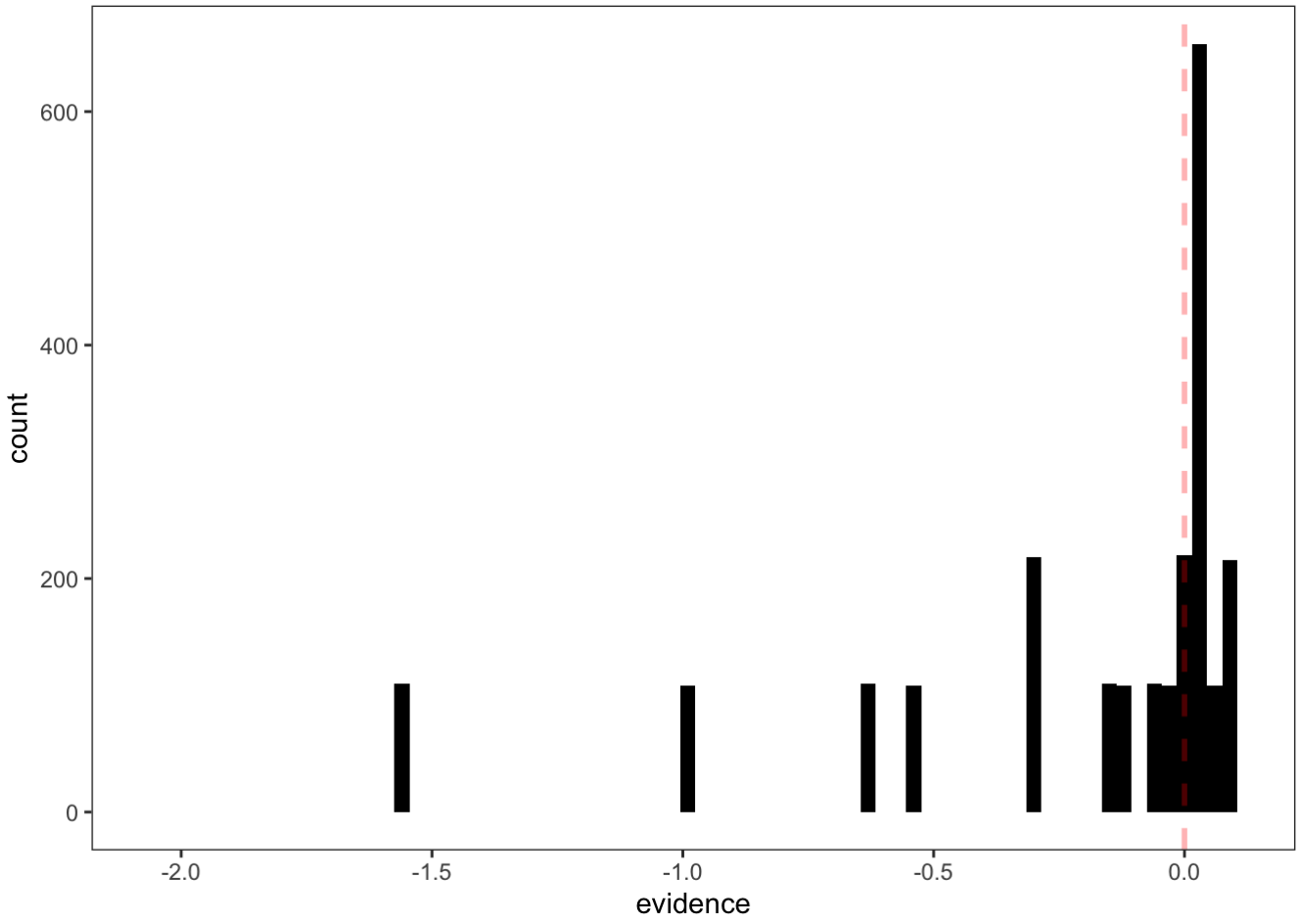
```
# preprocessing
responses_df <- full_df %>%
  rename( # rename to convert away from camel case
    worker_id = workerId,
    company_value = companyValue,
    ground_truth = groundTruth,
    p_contract_new = pContractNew,
    p_contract_old = pContractOld,
    p_superiority = pSup,
    start_time = startTime,
    resp_time = respTime,
    trial_dur = trialDur,
    trial_idx = trialIdx
  ) %>%
  filter(trial_idx != "practice", trial_idx != "mock") %>% # remove practice and mock
trials from responses dataframe, leave in full version
  mutate( # mutate to rows where intervene == -1 for some reason
    intervene = if_else(intervene == -1,
                        # repair
                        if_else((payoff == (contract_value - 1) | payoff == (-contrac
t_value - 1) | payoff == -1),
                                1, # payed for intervention
                                0), # didn't pay for intervention
                        # don't repair
                        as.numeric(intervene) # hack to avoid type error
                        )
  ) %>%
  mutate( # create ground truth metric for evidence in favor of decision
    evidence = log((p_contract_new - p_contract_old) / (1 / contract_value))
  )

head(responses_df)
```

```
## # A tibble: 6 x 28
##   worker_id batch condition baseline contract_value exchange
##   <chr>     <int> <chr>        <dbl>          <dbl>    <dbl>
## 1 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## 2 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## 3 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## 4 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## 5 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## 6 a3ee04d1     13 means_on…      0.5           2.25   0.0480
## # … with 22 more variables: start_gain_frame <chr>, total_bonus <dbl>,
## #   duration <dbl>, numeracy <int>, gender <chr>, age <chr>,
## #   education <chr>, chart_use <chr>, company_value <dbl>,
## #   ground_truth <dbl>, intervene <dbl>, outcome <chr>,
## #   p_contract_new <dbl>, p_contract_old <dbl>, p_superiority <int>,
## #   payoff <dbl>, resp_time <dbl>, start_time <dbl>, trial <chr>,
## #   trial_dur <dbl>, trial_idx <chr>, evidence <dbl>
```

We need the data in a format where it is prepared for modeling. This means that we want baseline as a factor rather than a numeric value.

```
# create data frame for model
model_df <- responses_df %>%
  mutate(
    baseline = as.factor(baseline),
    frame = as.factor(if_else(ground_truth > 0.5, "gain", "loss"))
  )

model_df %>% ggplot(aes(x = evidence)) +
  geom_histogram(fill = "black", binwidth = 0.03) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  xlim(quantile(model_df$evidence, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank())
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Distribution of Decisions

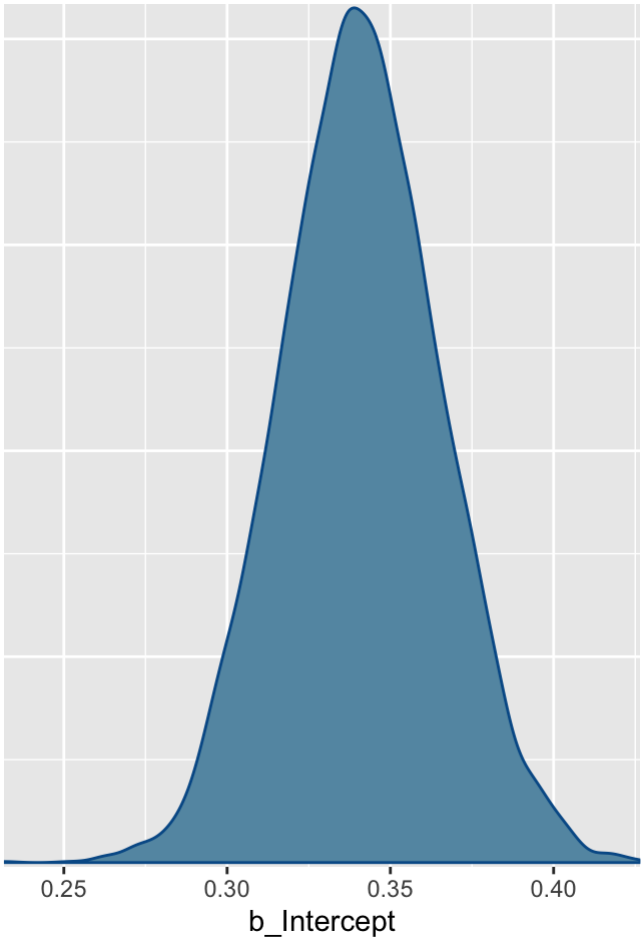
We start as simply as possible by just modeling the distribution of decisions using a probit link function and a linear model with only an intercept.

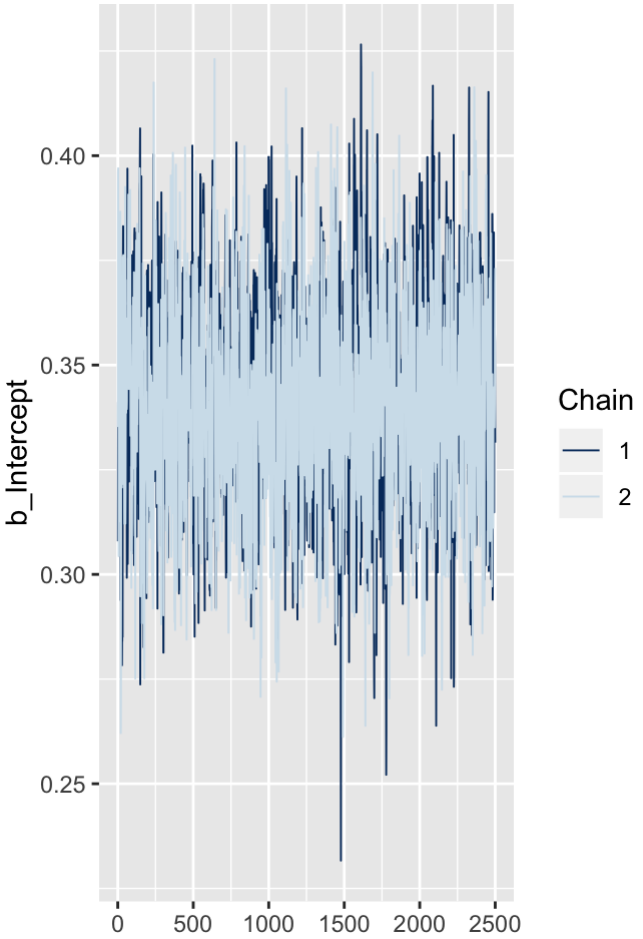
```
# get_prior(data = model_df,
#           family = bernoulli(link = "probit"),
#           formula = bf(intervene ~ 1))

# starting as simple as possible: learn the distribution of decisions
m.probit_intercept <- brm(data = model_df, family = bernoulli(link = "probit"),
              formula = bf(intervene ~ 1), #+ lf(disc ~ 1),
              prior = c(prior(normal(0, 1), class = Intercept)),
              iter = 3000, warmup = 500, chains = 2, cores = 2,
              file = "model-fits/probit_intercept_mdl")
```

Check diagnostics:

- Trace plots. The value of sigma seems pretty large.

```
# trace plots
plot(m.probit_intercept)
```





- Summary

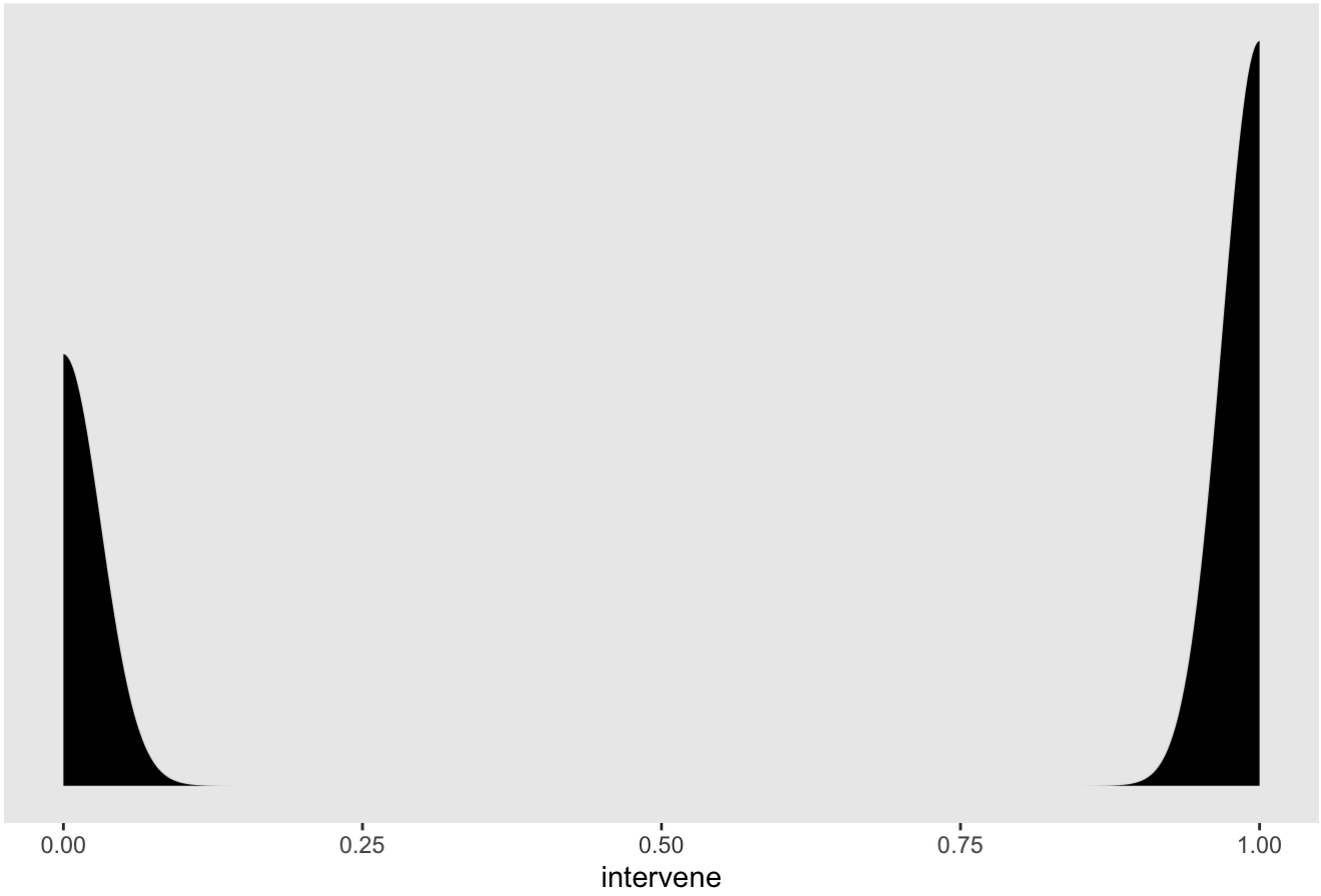
```
# model summary
print(m.probit_intercept)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ 1
##    Data: model_df_llo (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept     0.34      0.02     0.29     0.39       1628 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  select(evidence) %>% # this model should not be sensitive to evidence
  add_predicted_draws(m.probit_intercept, prediction = "intervene", seed = 1234, n =
200) %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

Posterior predictive distribution for intervention

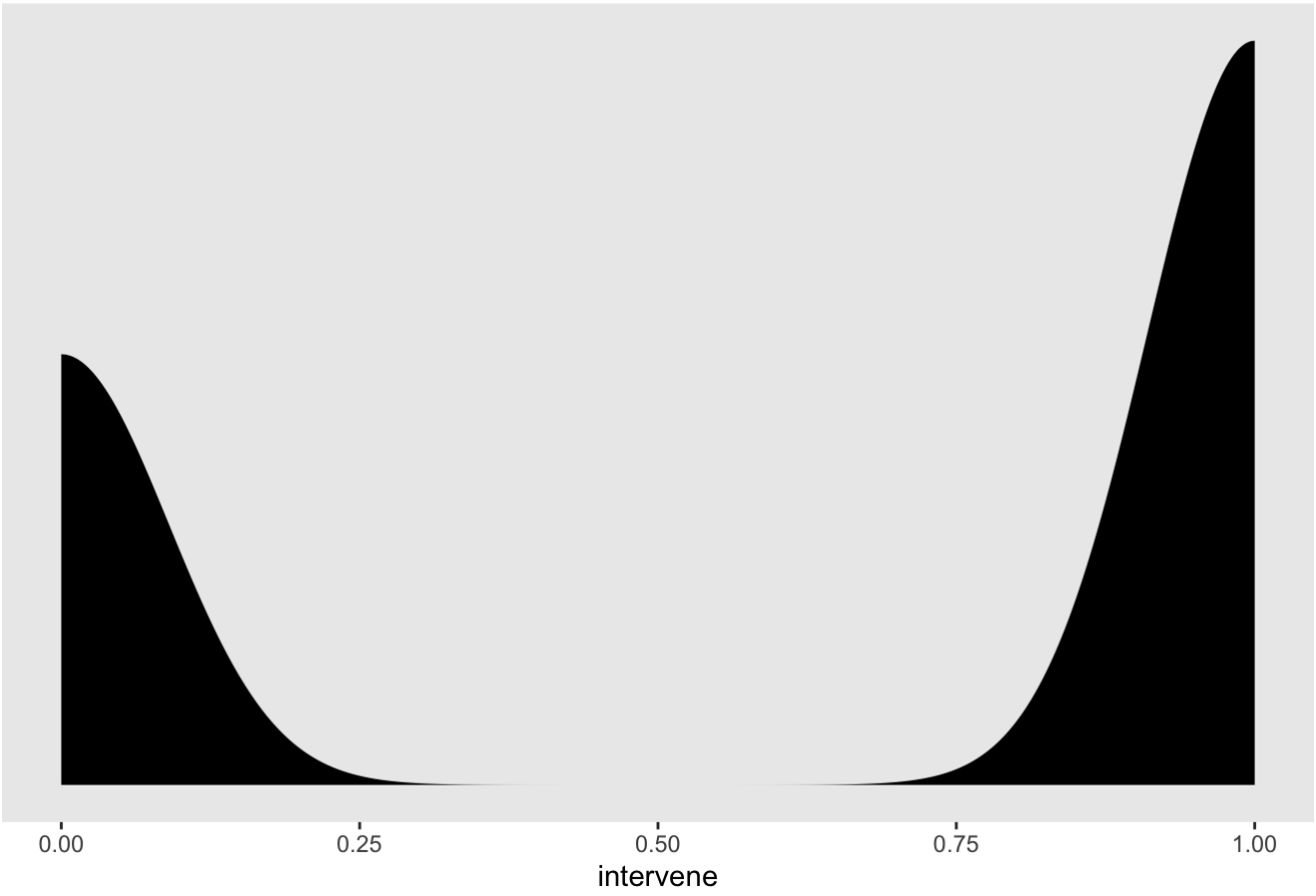


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

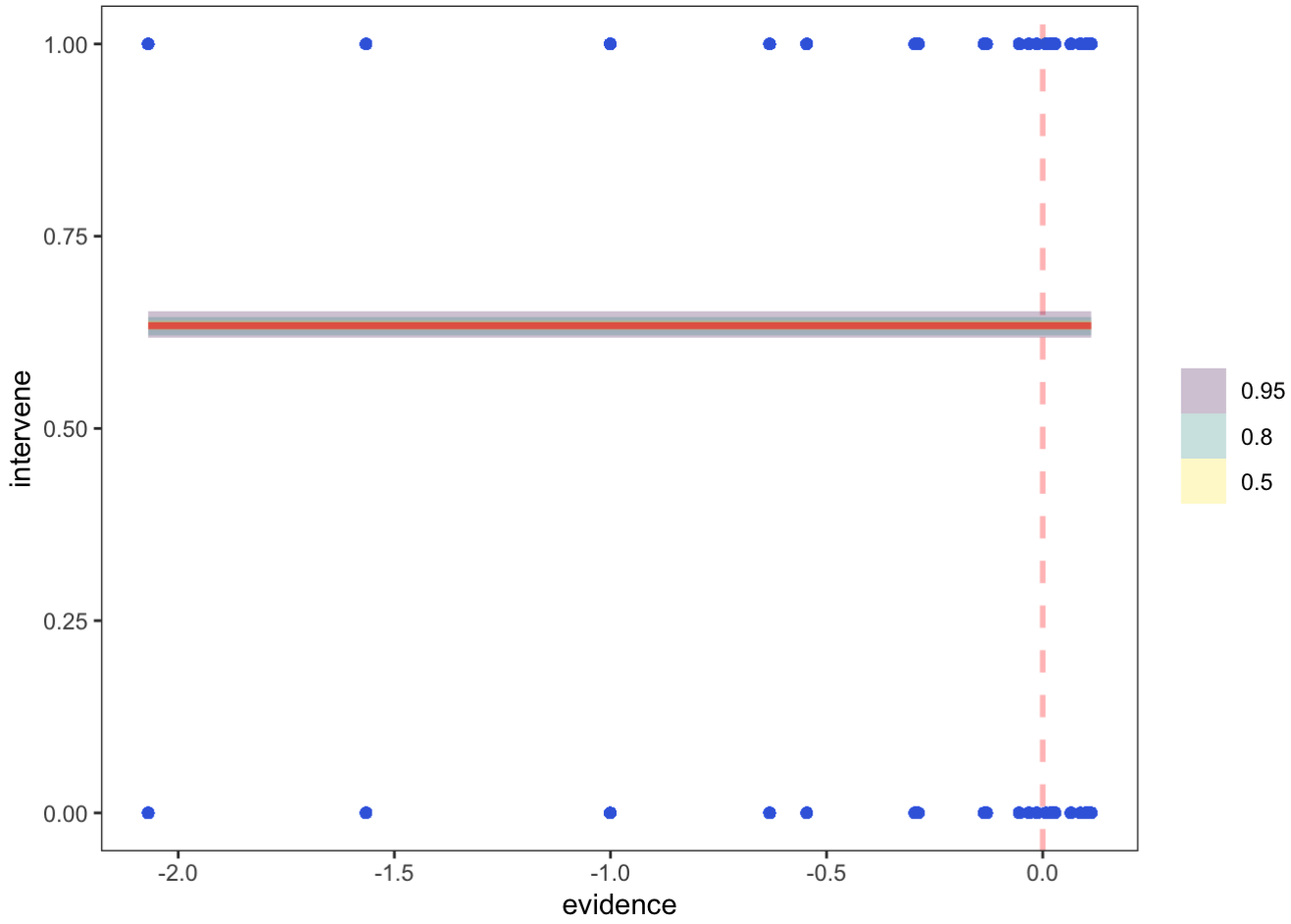
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric function. This should not have any slope.

```
model_df %>%
  # group_by(worker_id, trial_idx, intervene) %>%
  # data_grid(evidence = seq_range(evidence, n = 51)) %>%
  add_fitted_draws(m.probit_intercept, value = "pf", n = 200) %>%
  ggplot(aes(x = evidence, y = intervene)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  # geom_line(aes(y = pf, group = .draw)) +
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15, color = "royalblue") +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank())
```



Linear Model with Probit Link Function

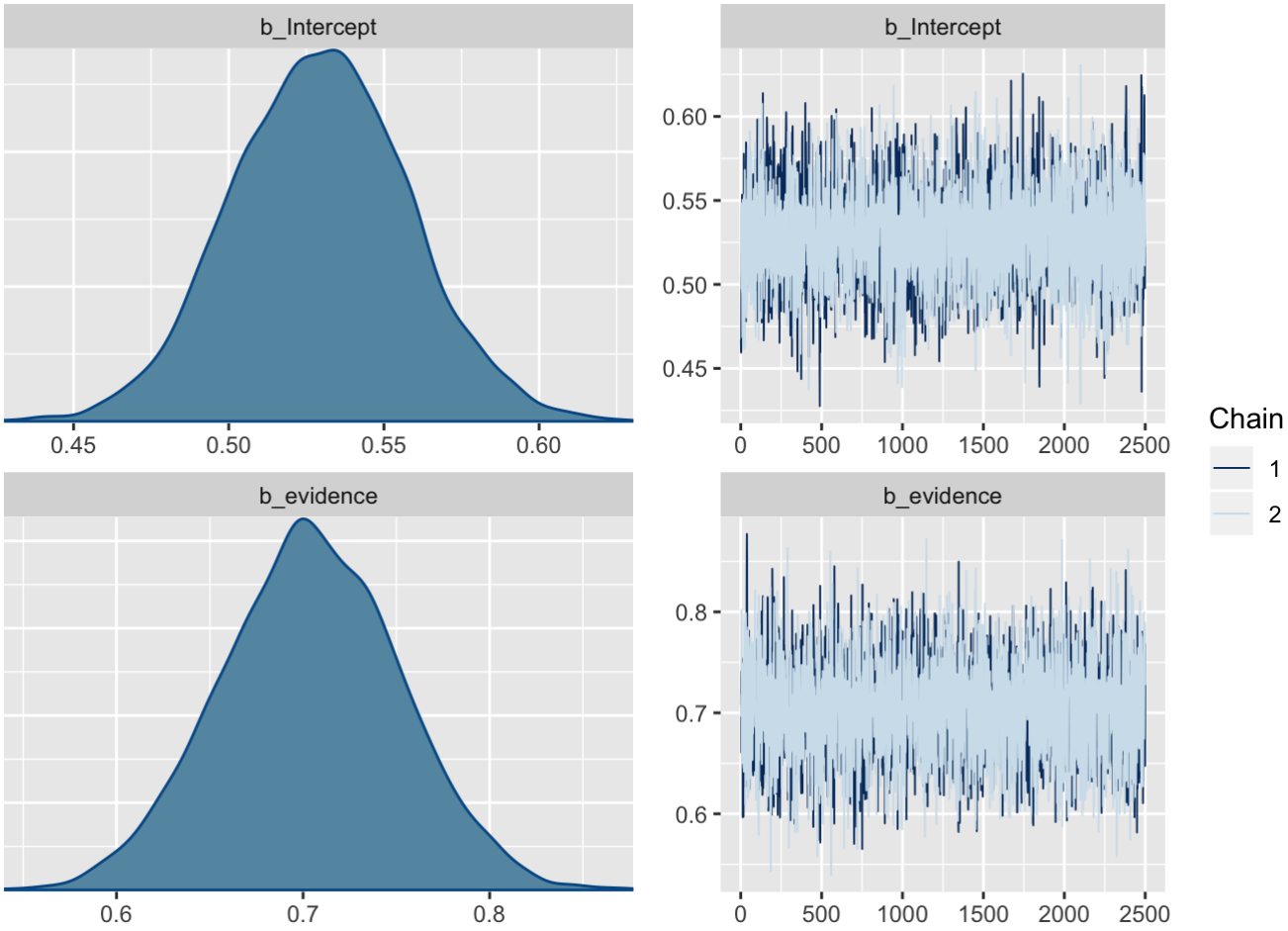
Now well add a slope parameter to our model to make it a simple linear model where decisions to intervene are a function of the probability of getting the contract with the new machine.

```
# get_prior(data = model_df,
#           family = binomial("probit"),
#           intervene | trials(1) ~ 1 + evidence)

# linear model with probit link
m.probit <- brm(data = model_df, family = bernoulli(link = "probit"),
                formula = bf(intervene ~ 1 + evidence),
                prior = c(prior(normal(0, 1), class = Intercept),
                          prior(normal(0, 1), class = b)),
                iter = 3000, warmup = 500, chains = 2, cores = 2,
                file = "model-fits/probit_mdl")
```

Check diagnostics:

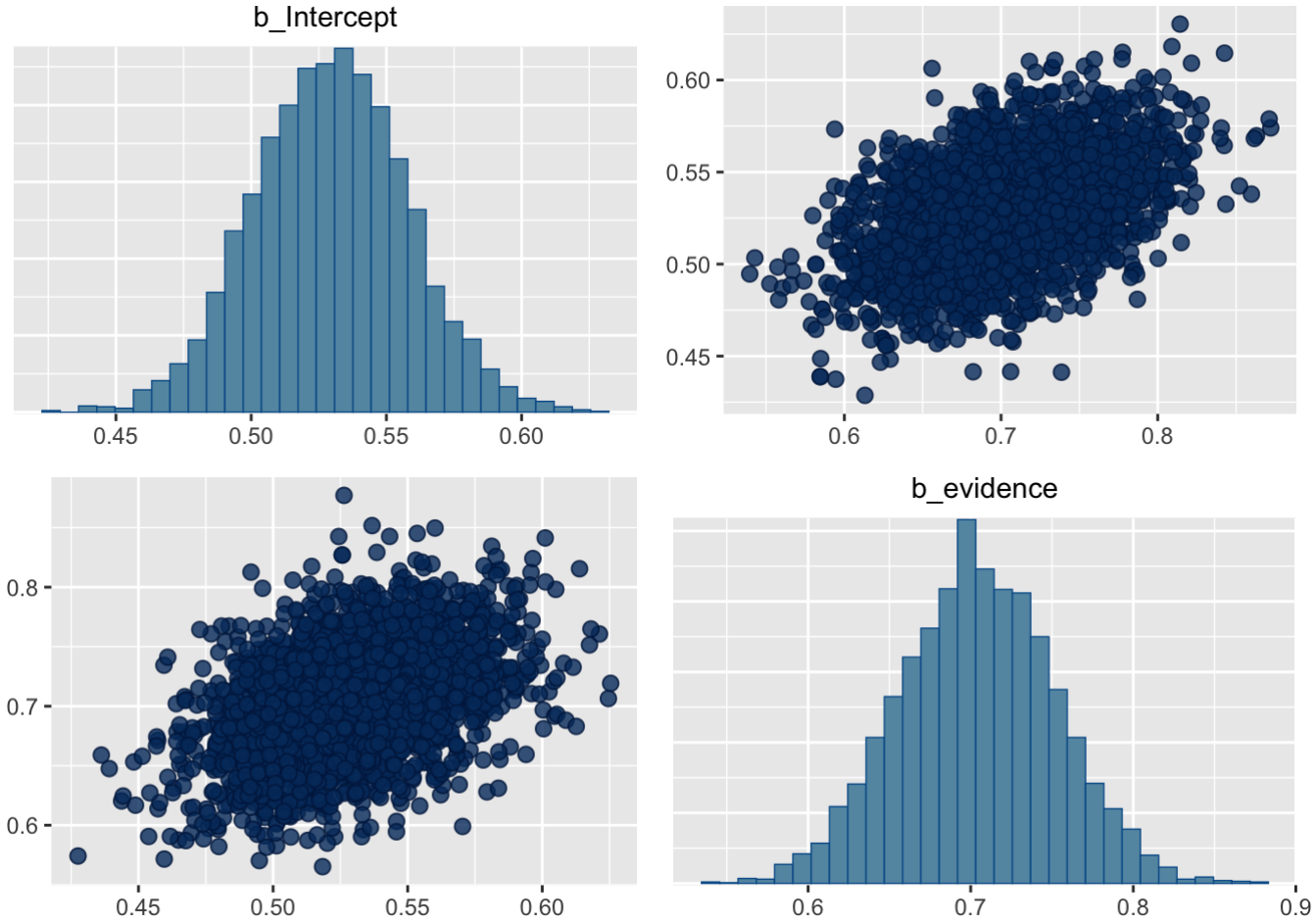
- Trace plots

```
# trace plots
plot(m.probit)
```



- Pairs plot.

```
# pairs plot
pairs(m.probit)
```



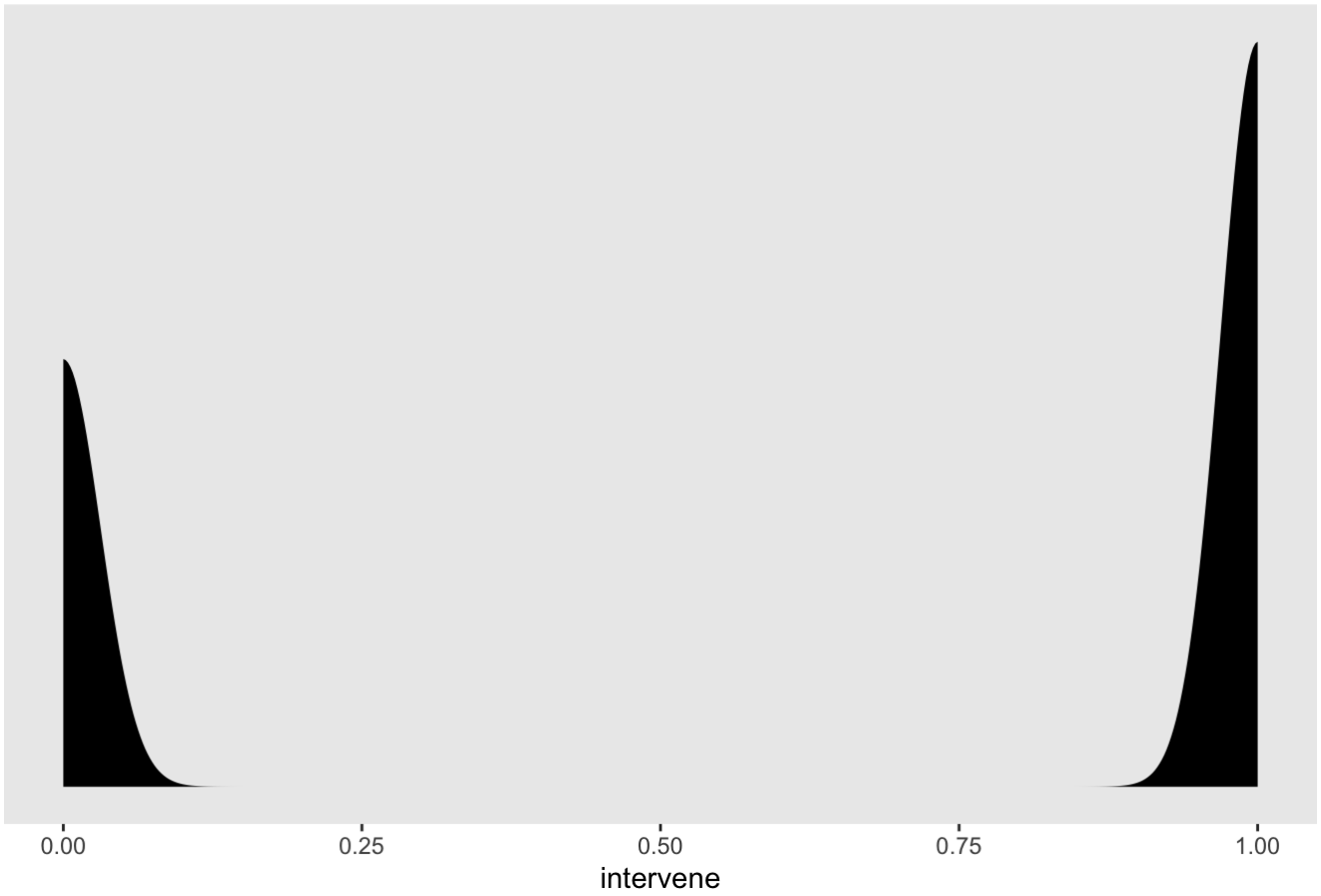
- Summary

```
# model summary
print(m.probit)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ 1 + evidence
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept     0.53      0.03     0.47     0.59       3900 1.00
## evidence      0.70      0.05     0.61     0.80       4338 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  select(evidence) %>%
  add_predicted_draws(m.probit, prediction = "intervene", seed = 1234, n = 200) %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

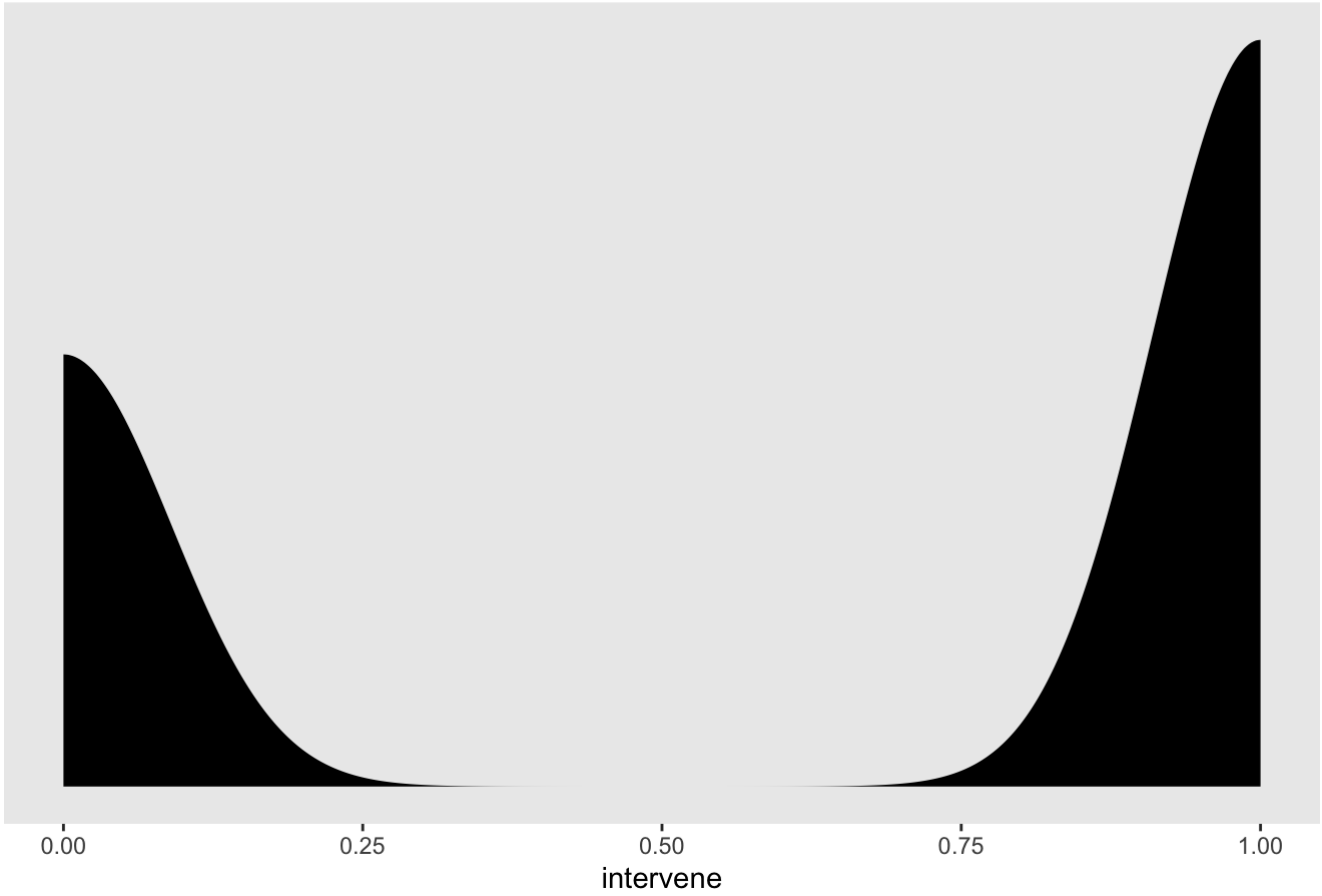


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

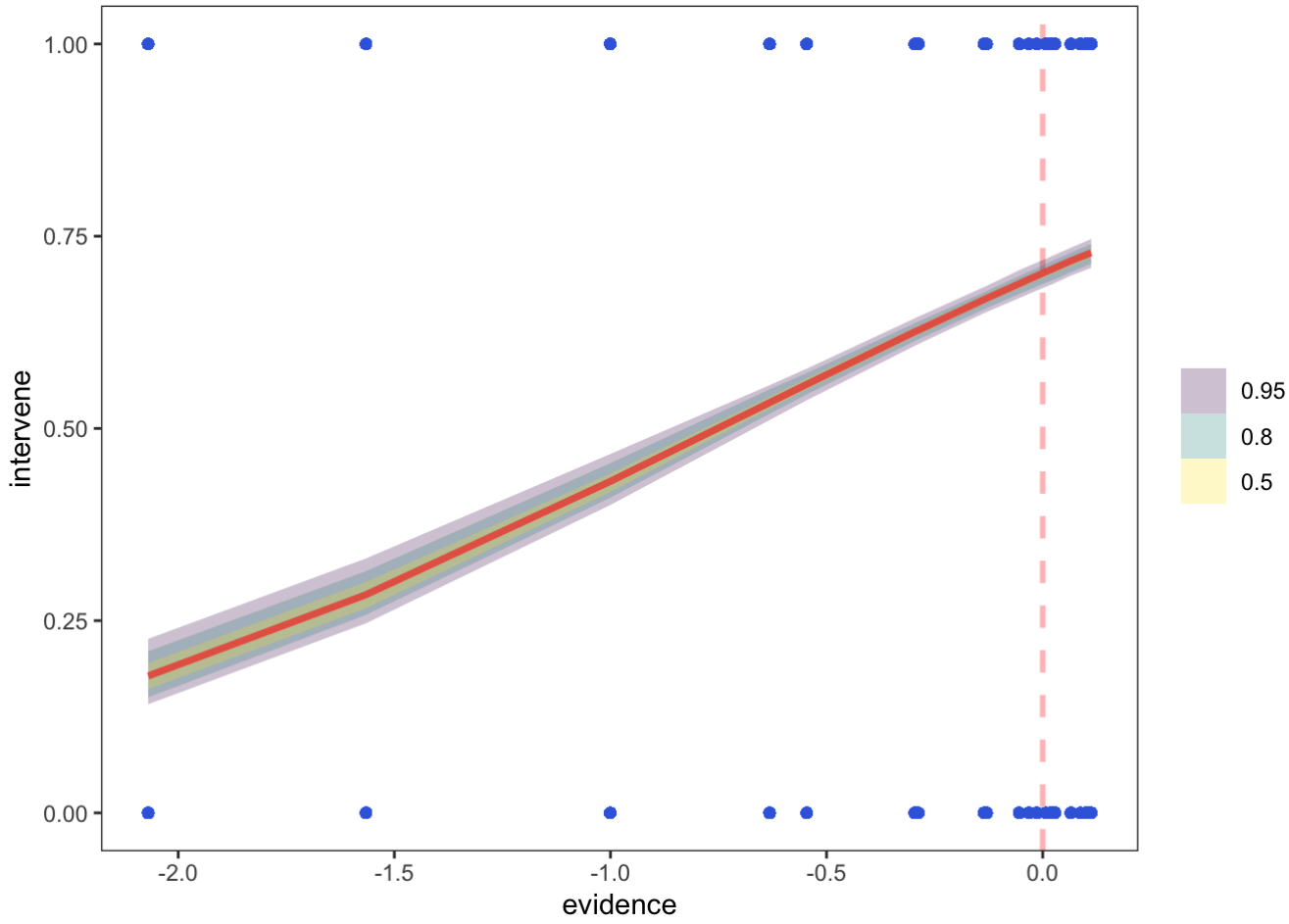
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric function.

```
model_df %>%
  # group_by(worker_id, trial_idx, intervene) %>%
  # data_grid(evidence = seq_range(evidence, n = 51)) %>%
  add_fitted_draws(m.probit, value = "pf", n = 200) %>%
  ggplot(aes(x = evidence, y = intervene)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  # geom_line(aes(y = pf, group = .draw)) +
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15, color = "royalblue") +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank())
```



Add Different Linear Models per Visualization Condition

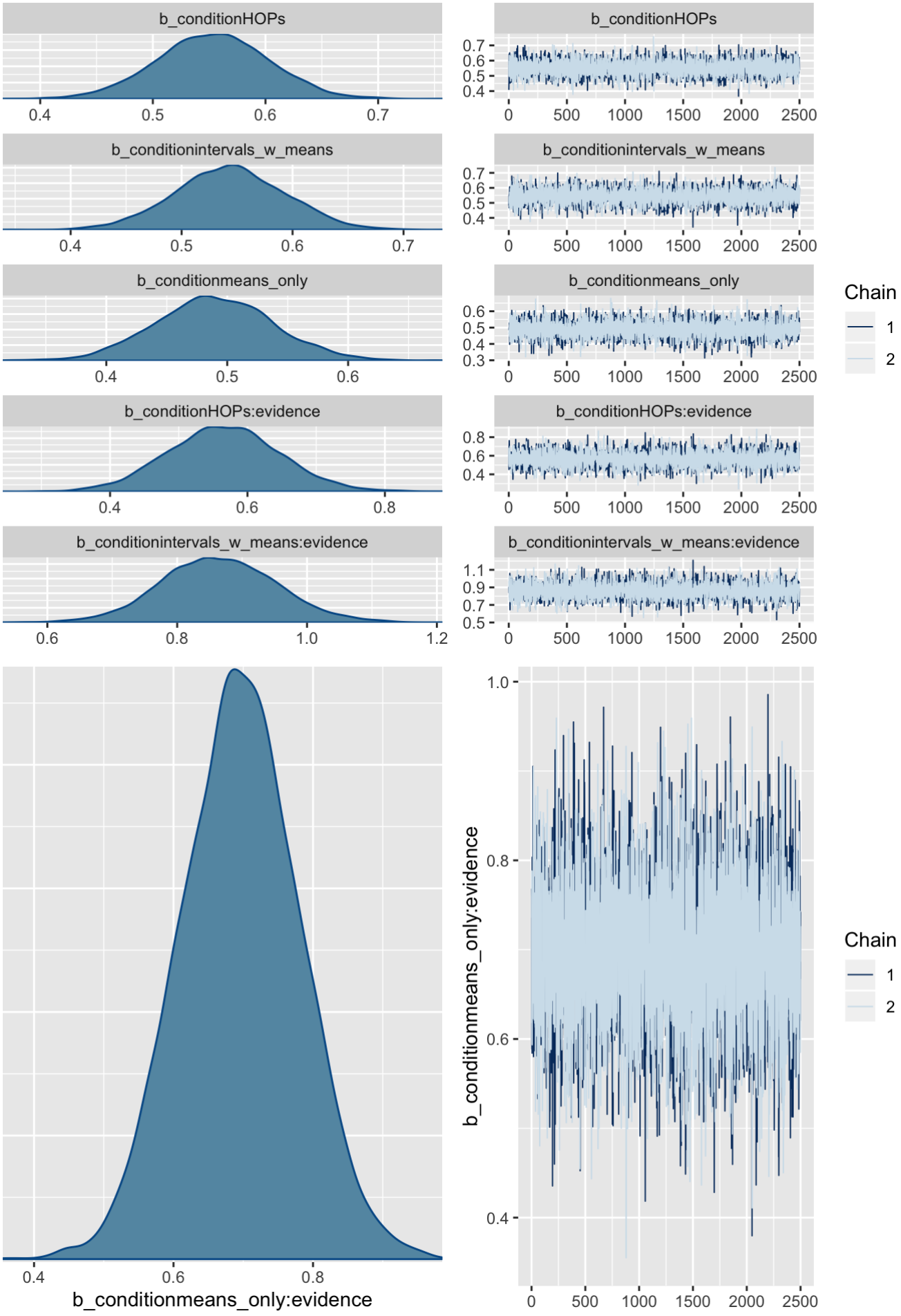
Now we add visualization condition as a predictor of both the point of subjective equality and the slope of the psychometric function.

```
# get_prior(data = model_df, family = bernoulli(link = "probit"),
#           formula = bf(intervene ~ 0 + condition + evidence:condition))

# linear model with probit link
m.vis.probit <- brm(data = model_df, family = bernoulli(link = "probit"),
              formula = bf(intervene ~ 0 + condition + evidence:condition),
              prior = c(prior(normal(0, 1), class = b)),
              iter = 3000, warmup = 500, chains = 2, cores = 2,
              file = "model-fits/probit_mdl_vis")
```

Check diagnostics:

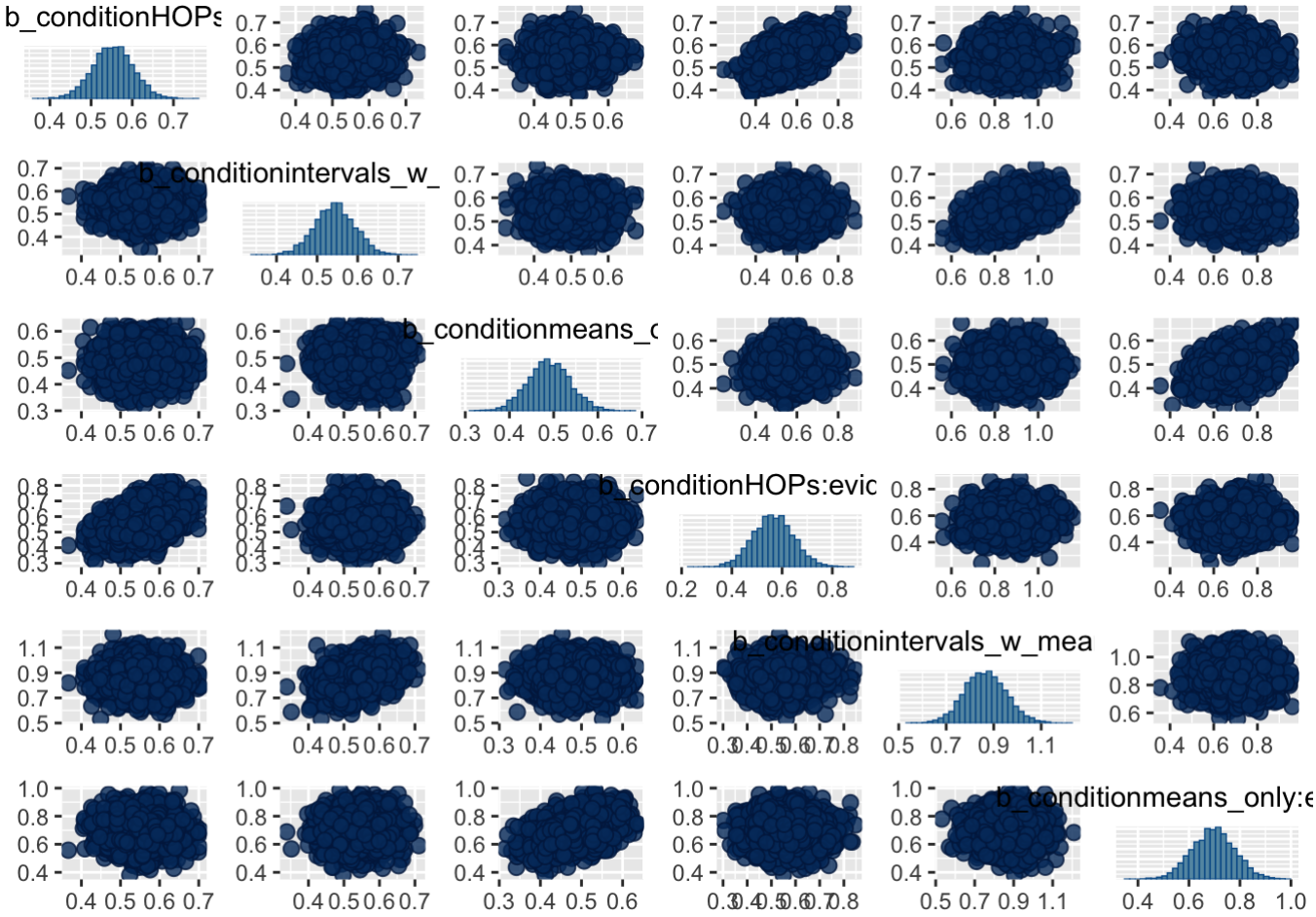
- Trace plots

```
# trace plots
plot(m.vis.probit)
```


b_conditionHOPs
b_conditionintervals_w_means
b_conditionmeans_only
b_conditionHOPs:evidence
b_conditionintervals_w_means:evidence
b_conditionmeans_only:evidence
Chain
1
2

- Pairs plot

```
# pairs plot
pairs(m.vis.probit)
```



- Summary

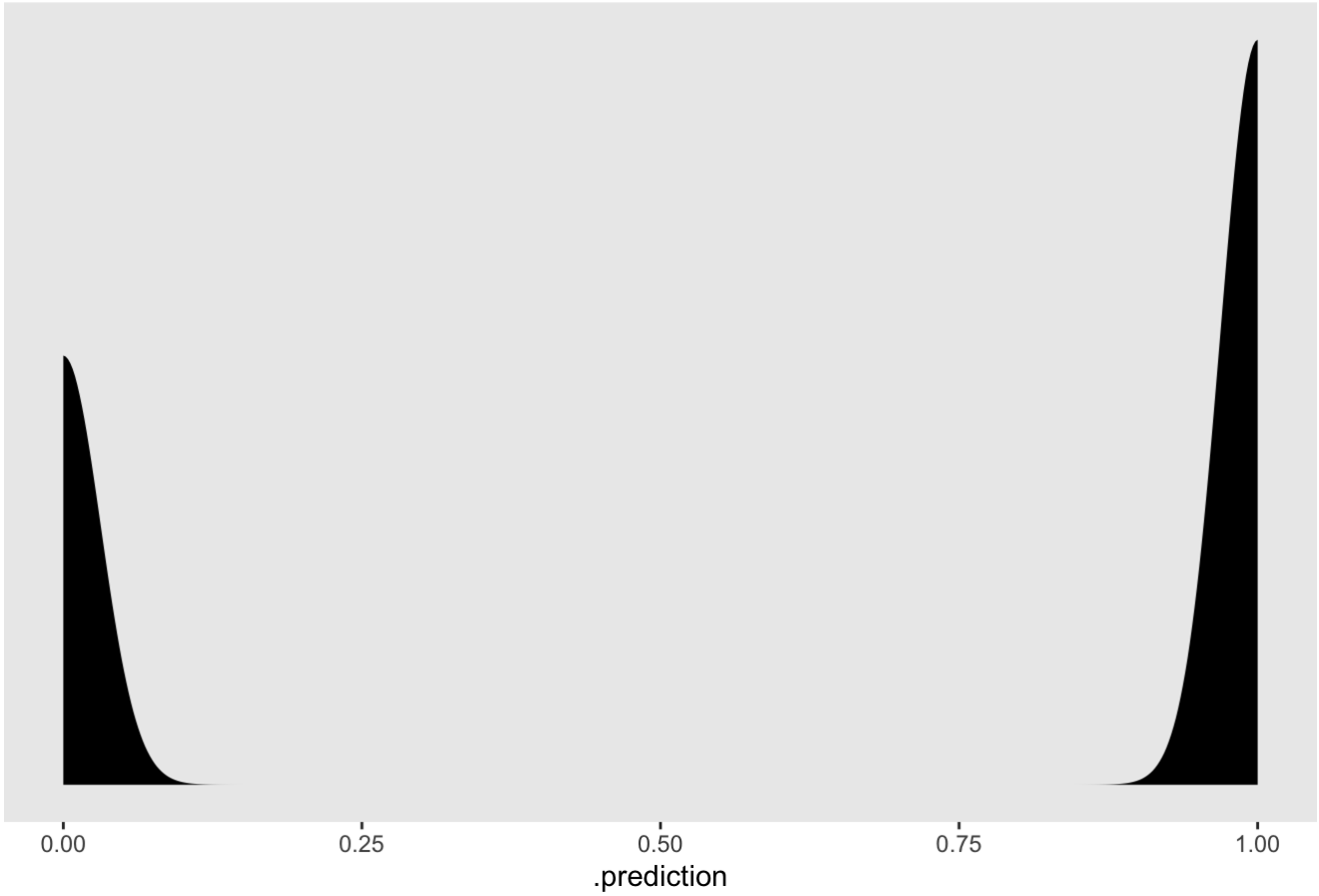
```
# model summary
print(m.vis.probit)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ 0 + condition + evidence:condition
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Population-Level Effects:
##                                    Estimate Est.Error l-95% CI u-95% CI
## conditionHOPs                          0.55      0.05     0.46     0.65
## conditionintervals_w_means             0.54      0.05     0.45     0.64
## conditionmeans_only                    0.49      0.05     0.40     0.59
## conditionHOPs:evidence                 0.57      0.08     0.41     0.73
## conditionintervals_w_means:evidence    0.86      0.09     0.69     1.04
## conditionmeans_only:evidence           0.70      0.08     0.53     0.86
##                                    Eff.Sample Rhat
## conditionHOPs                            5453 1.00
## conditionintervals_w_means               5644 1.00
## conditionmeans_only                      5842 1.00
## conditionHOPs:evidence                   5057 1.00
## conditionintervals_w_means:evidence      5684 1.00
## conditionmeans_only:evidence             5926 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  group_by(condition, evidence) %>%
  add_predicted_draws(m.vis.probit, seed = 1234, n = 200) %>%
  ggplot(aes(x = .prediction)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

Posterior predictive distribution for intervention

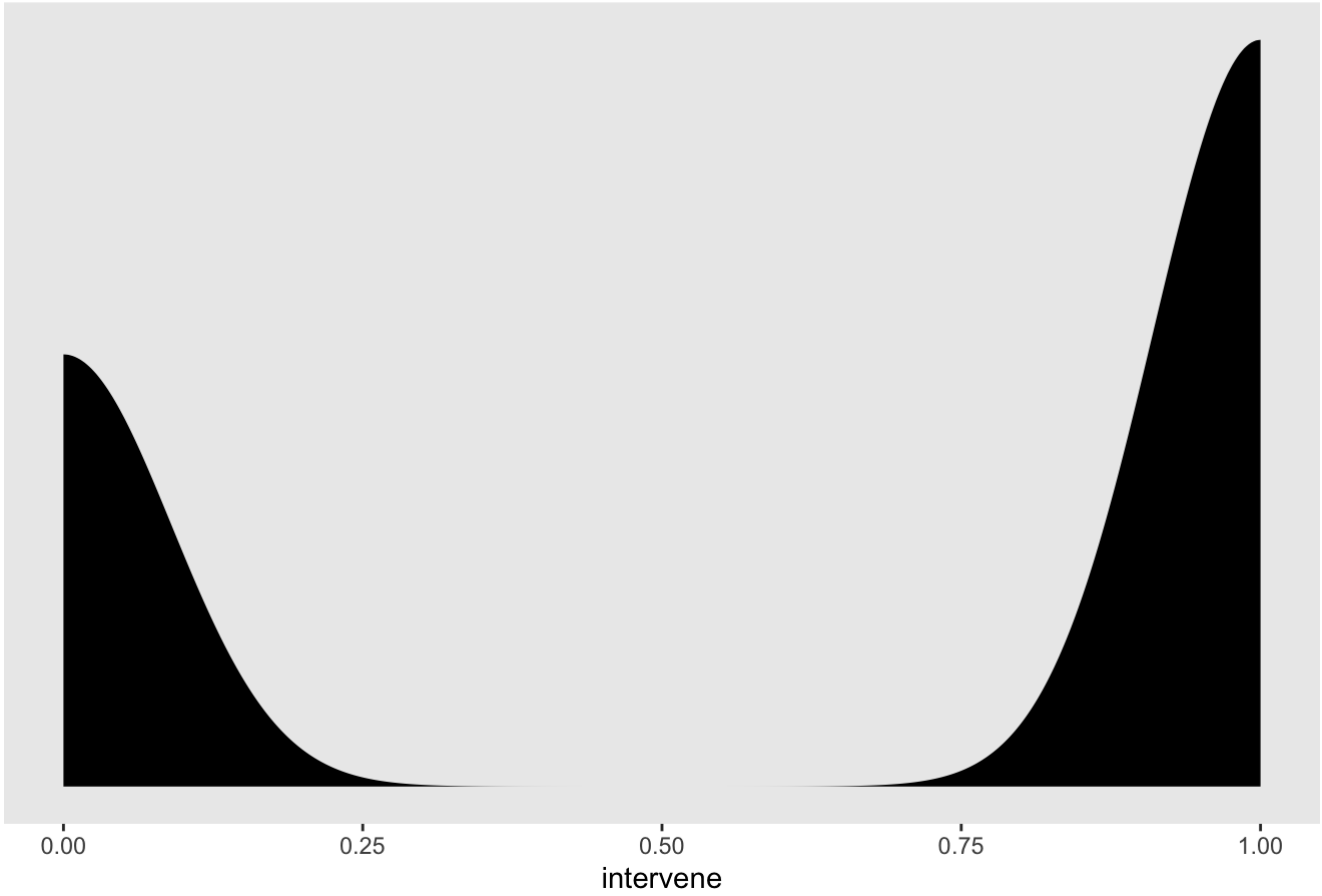


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

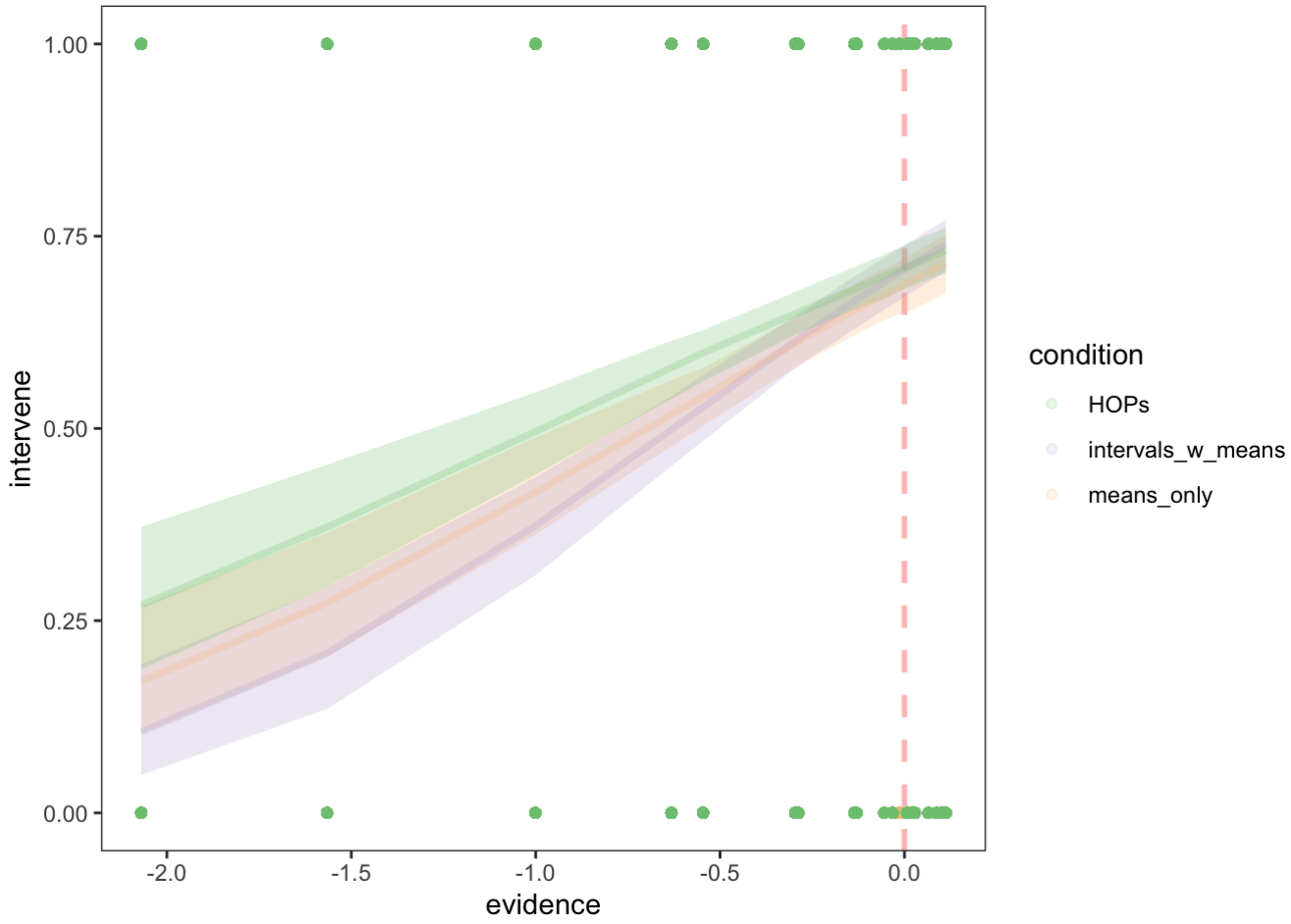
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric functions.

```
model_df %>%
  add_fitted_draws(m.vis.probit, value = "pf", n = 200) %>%
  ggplot(aes(x = evidence, y = intervene, color = condition)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  # geom_line(aes(y = pf, group = .draw)) +
  stat_lineribbon(aes(y = pf, fill = condition), .width = .95, alpha = .25, show.lege
nd = FALSE) +
  geom_point(alpha = .15) +
  scale_fill_brewer(type = "qual", palette = 1) +
  scale_color_brewer(type = "qual", palette = 1) +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank())
```



Add Hierarchy for Slopes and Intercepts

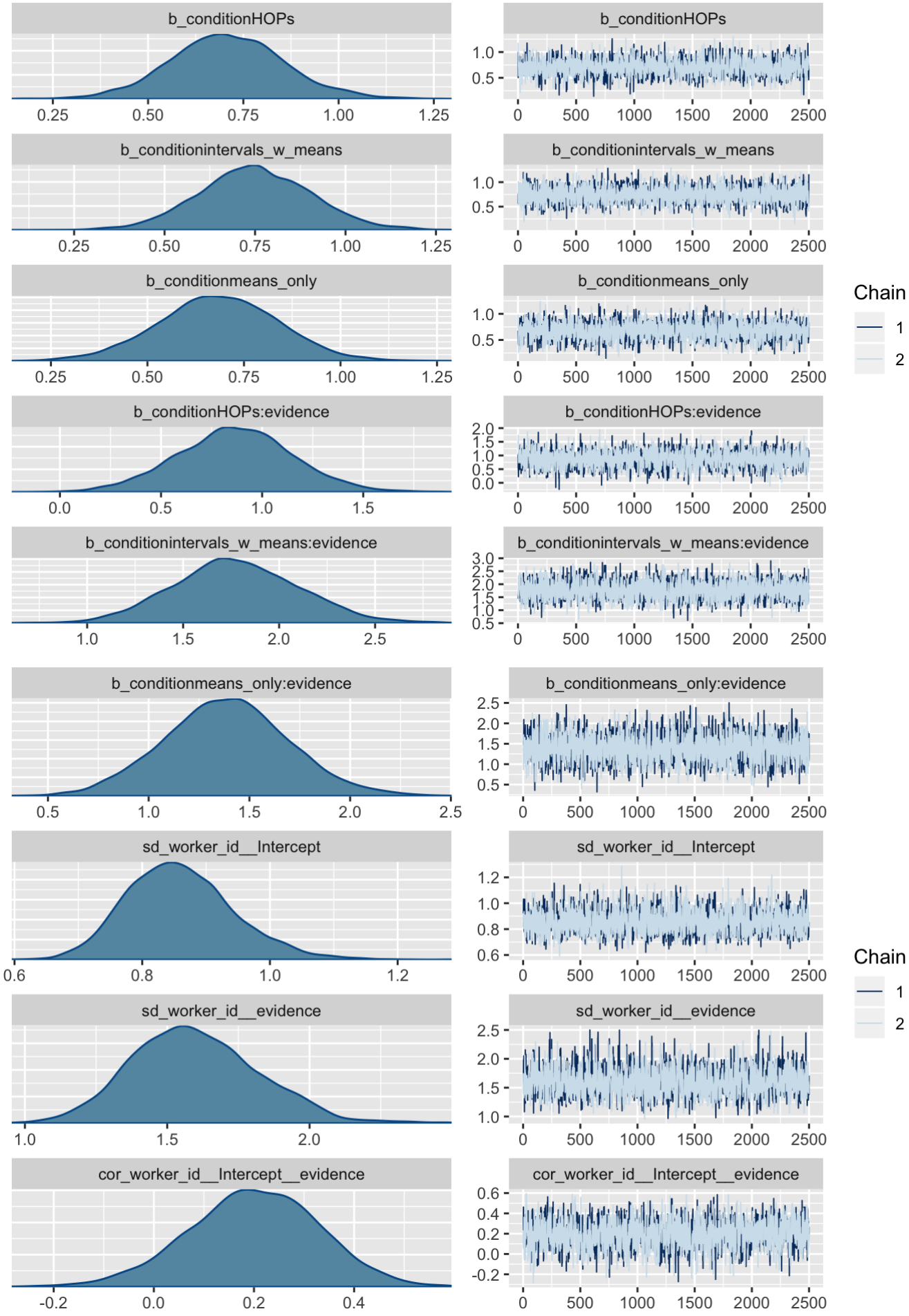
The models we've created thus far fail to account for much of the noise in the data. Here, we attempt to parse some heterogeniety in responses by modeling a random effect of worker on slopes and intercepts. This introduces a hierarchical component to our model in order to account for individual differences in the best fitting linear model for each worker's data.

```
# get_prior(data = model_df, family = bernoulli(link = "probit"),
#           formula = bf(intervene ~ 0 + condition + evidence:condition))

# linear model with probit link
m.vis.wrkr.probit <- brm(data = model_df, family = bernoulli(link = "probit"),
              formula = bf(intervene ~ (1 + evidence|worker_id) + 0 + condition + evi
dence:condition),
              prior = c(prior(normal(0, 1), class = b)),
              iter = 3000, warmup = 500, chains = 2, cores = 2,
              file = "model-fits/probit_mdl_vis_wrkr")
```

Check diagnostics:

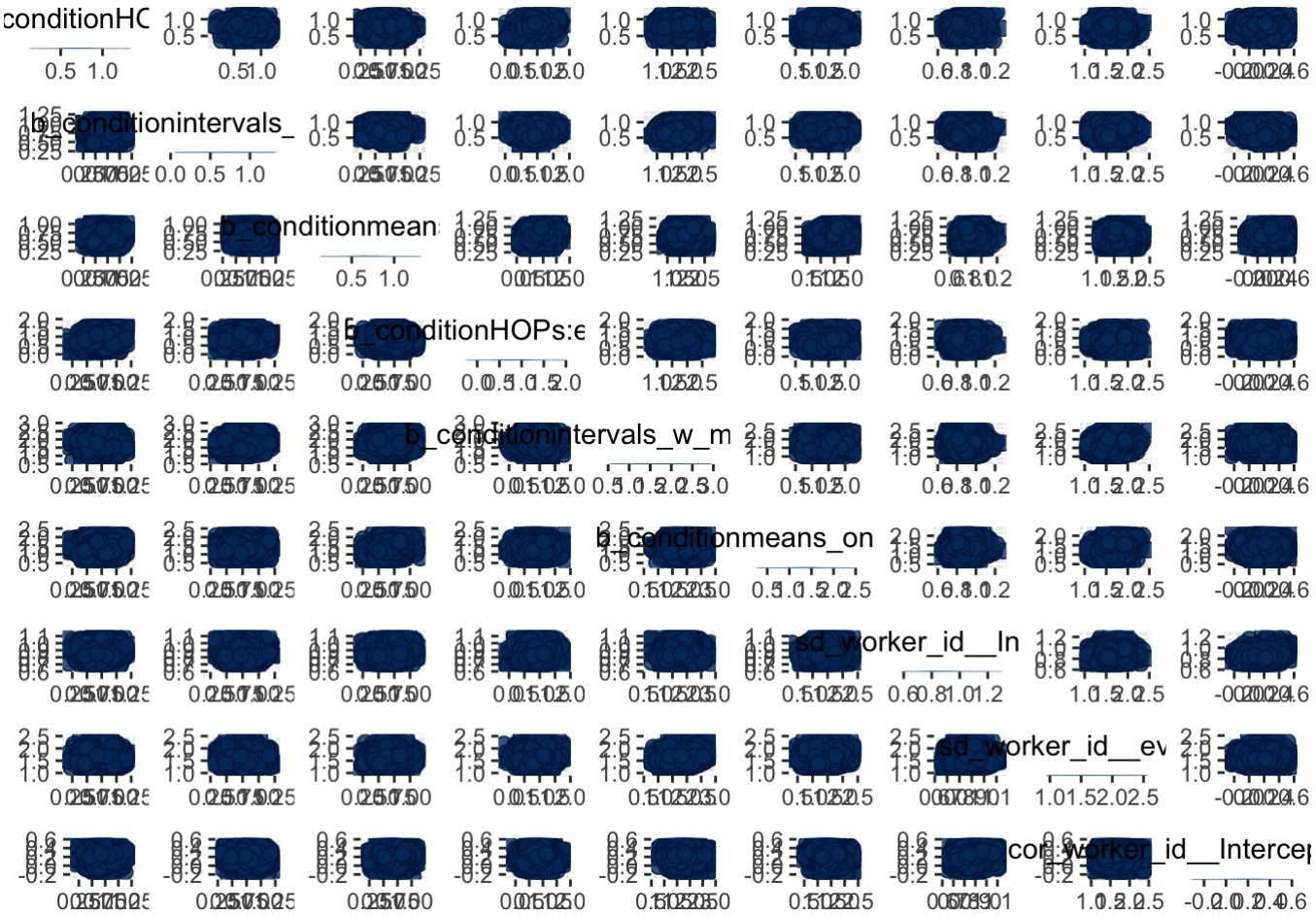
- Trace plots

```
# trace plots
plot(m.vis.wrkr.probit)
```


b_conditionHOPs
b_conditionintervals_w_means
b_conditionmeans_only
b_conditionHOPs:evidence
b_conditionintervals_w_means:evidence
b_conditionmeans_only:evidence
sd_worker_id__Intercept
sd_worker_id__evidence
cor_worker_id__Intercept__evidence
Chain
1
2

- Pairs plot

```
# pairs plot
pairs(m.vis.wrkr.probit)
```



- Summary

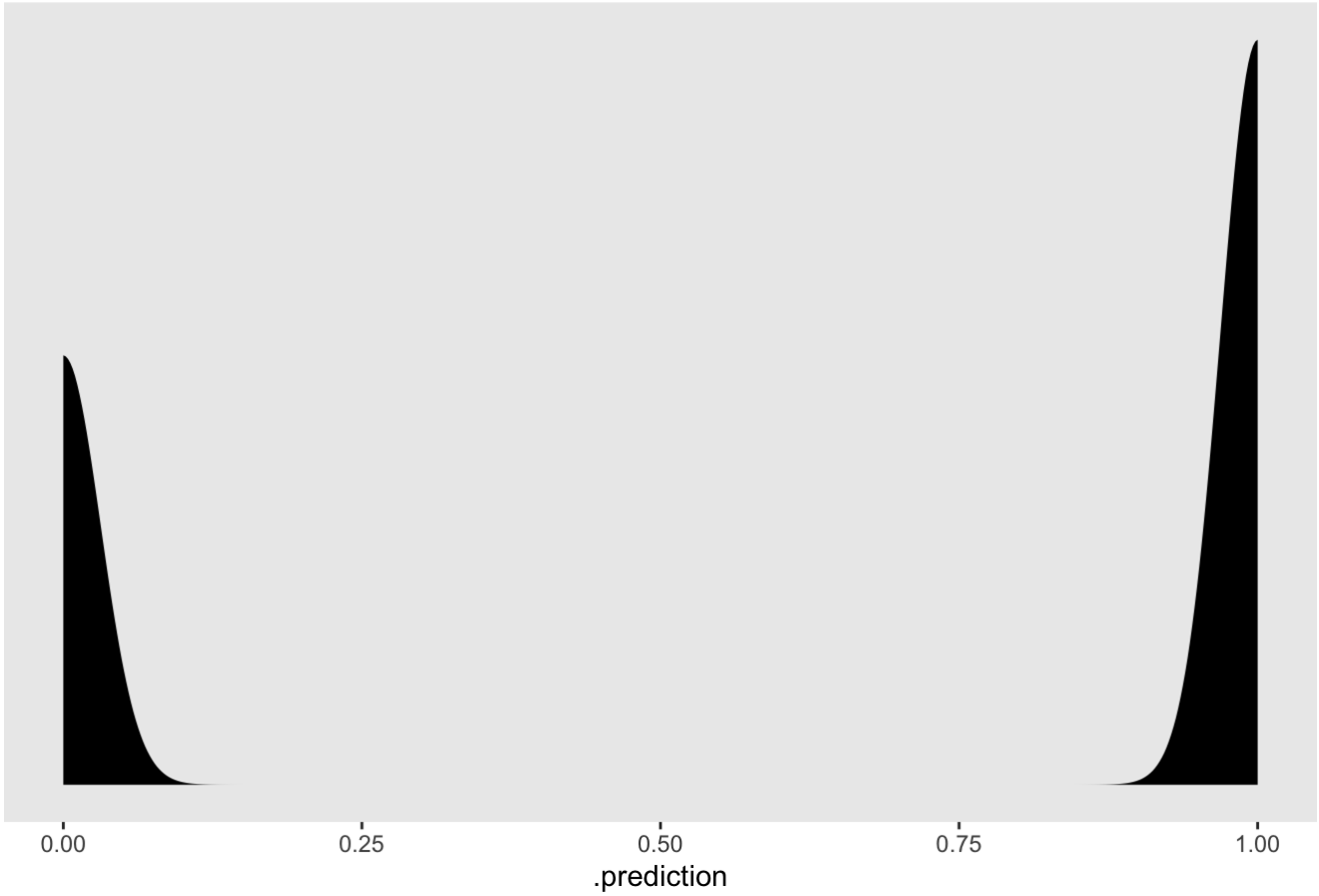
```
# model summary
print(m.vis.wrkr.probit)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ (1 + evidence | worker_id) + 0 + condition + evidence:conditi
on
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Group-Level Effects:
## ~worker_id (Number of levels: 109)
##                         Estimate Est.Error l-95% CI u-95% CI Eff.Sample
## sd(Intercept)               0.86      0.08     0.71     1.04       1585
## sd(evidence)                1.60      0.22     1.20     2.05       1434
## cor(Intercept,evidence)     0.20      0.13    -0.06     0.45       1253
##                         Rhat
## sd(Intercept)           1.00
## sd(evidence)            1.00
## cor(Intercept,evidence) 1.00
##
## Population-Level Effects:
##                                    Estimate Est.Error l-95% CI u-95% CI
## conditionHOPs                          0.71      0.15     0.41     1.02
## conditionintervals_w_means             0.75      0.15     0.45     1.06
## conditionmeans_only                    0.68      0.15     0.38     0.98
## conditionHOPs:evidence                 0.87      0.29     0.30     1.43
## conditionintervals_w_means:evidence    1.78      0.33     1.16     2.43
## conditionmeans_only:evidence           1.39      0.31     0.78     2.01
##                                    Eff.Sample Rhat
## conditionHOPs                            1982 1.00
## conditionintervals_w_means               2455 1.00
## conditionmeans_only                      2204 1.00
## conditionHOPs:evidence                   2346 1.00
## conditionintervals_w_means:evidence      2441 1.00
## conditionmeans_only:evidence             2508 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  group_by(worker_id, condition, evidence) %>%
  add_predicted_draws(m.vis.wrkr.probit, seed = 1234, n = 200) %>%
  ggplot(aes(x = .prediction)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

Posterior predictive distribution for intervention

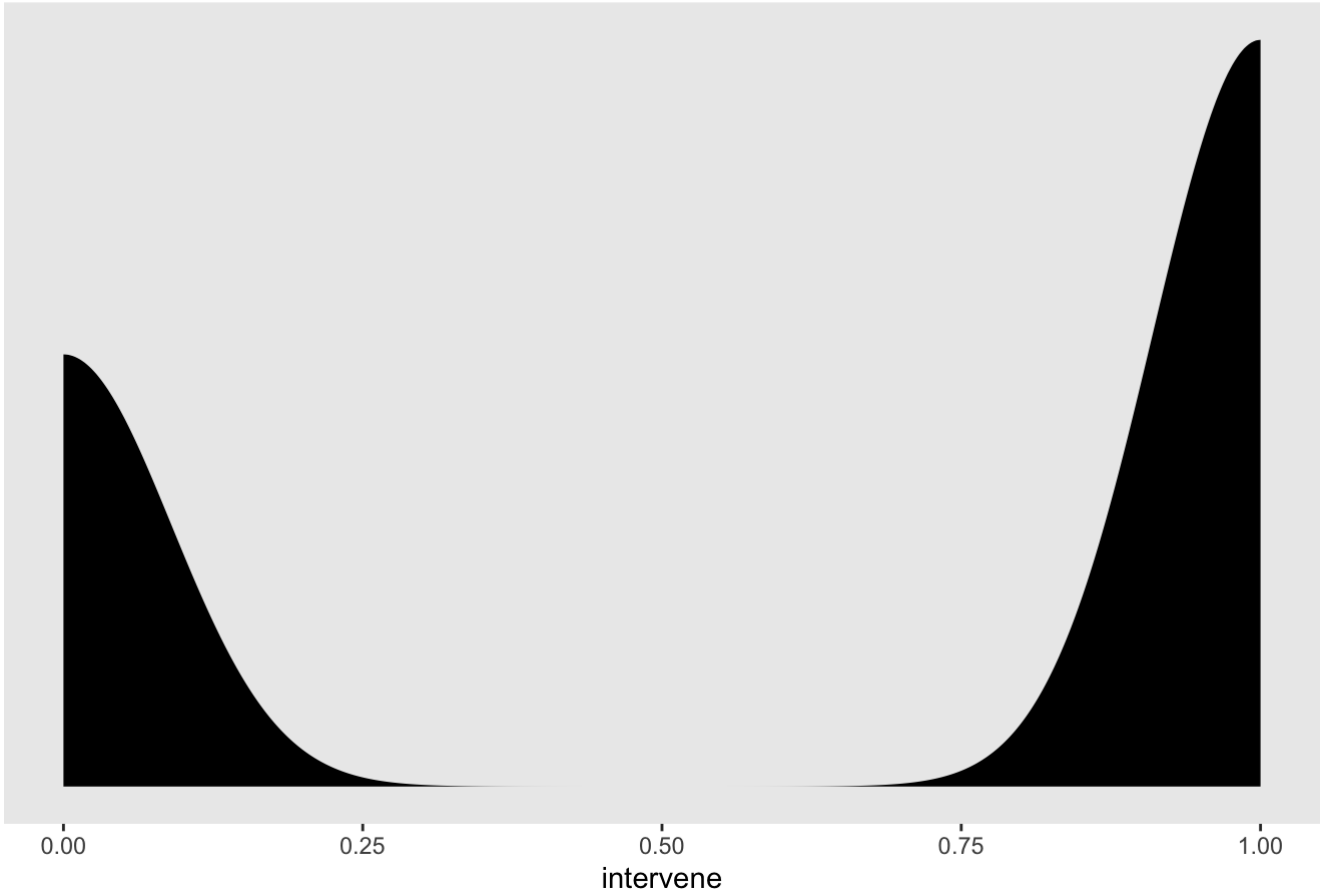


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

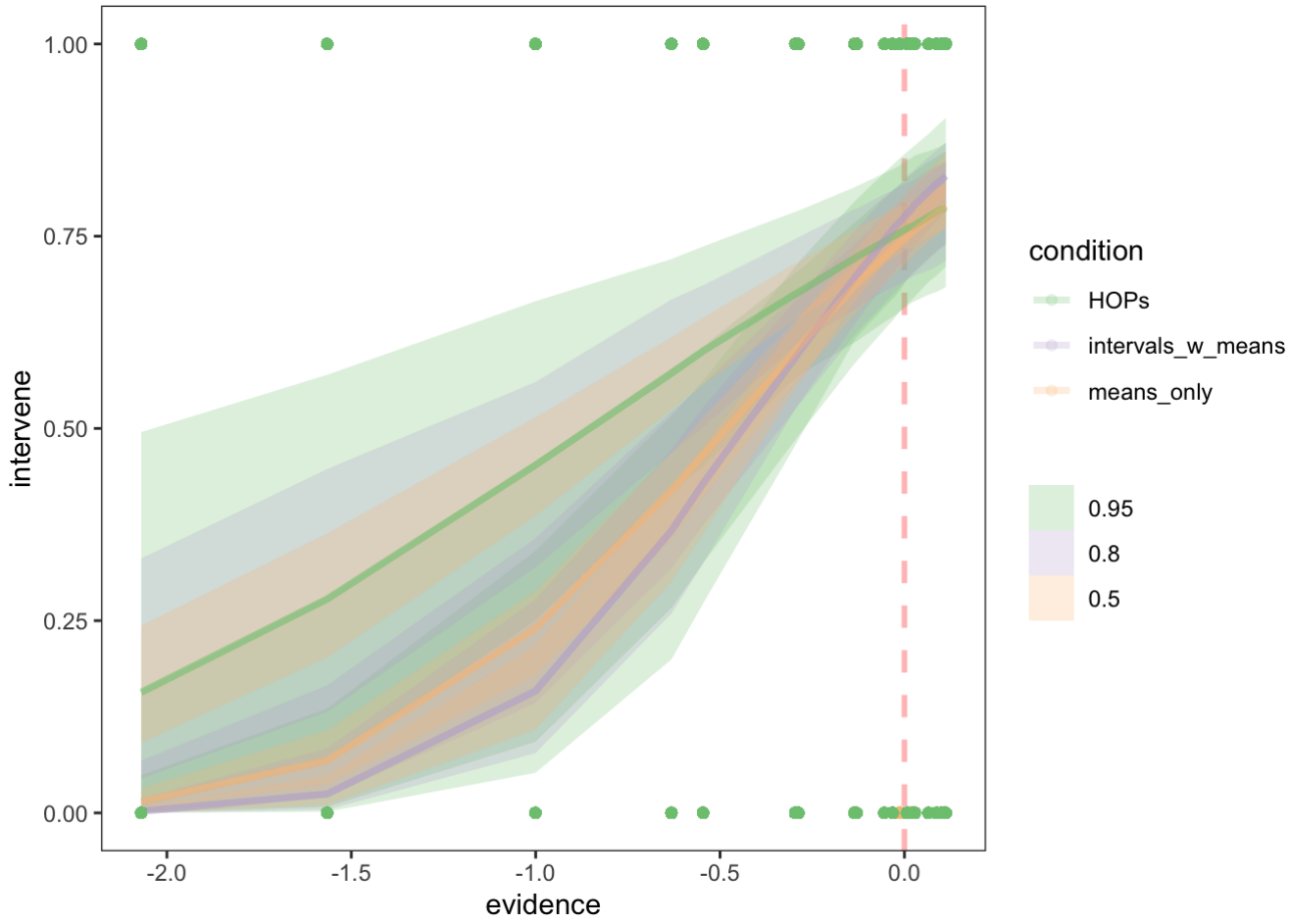
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric function in each condition for the average observer.

```
model_df %>%
  group_by(evidence, condition, worker_id) %>%
  add_fitted_draws(m.vis.wrkr.probit, value = "pf", re_formula = NA, n = 200) %>%
  ggplot(aes(x = evidence, y = intervene, color = condition)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15) +
  scale_fill_brewer(type = "qual", palette = 1) +
  scale_color_brewer(type = "qual", palette = 1) +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank())
```



Now we're seeing more separation between visualization conditions.

Add a Predictors for Baseline Condition

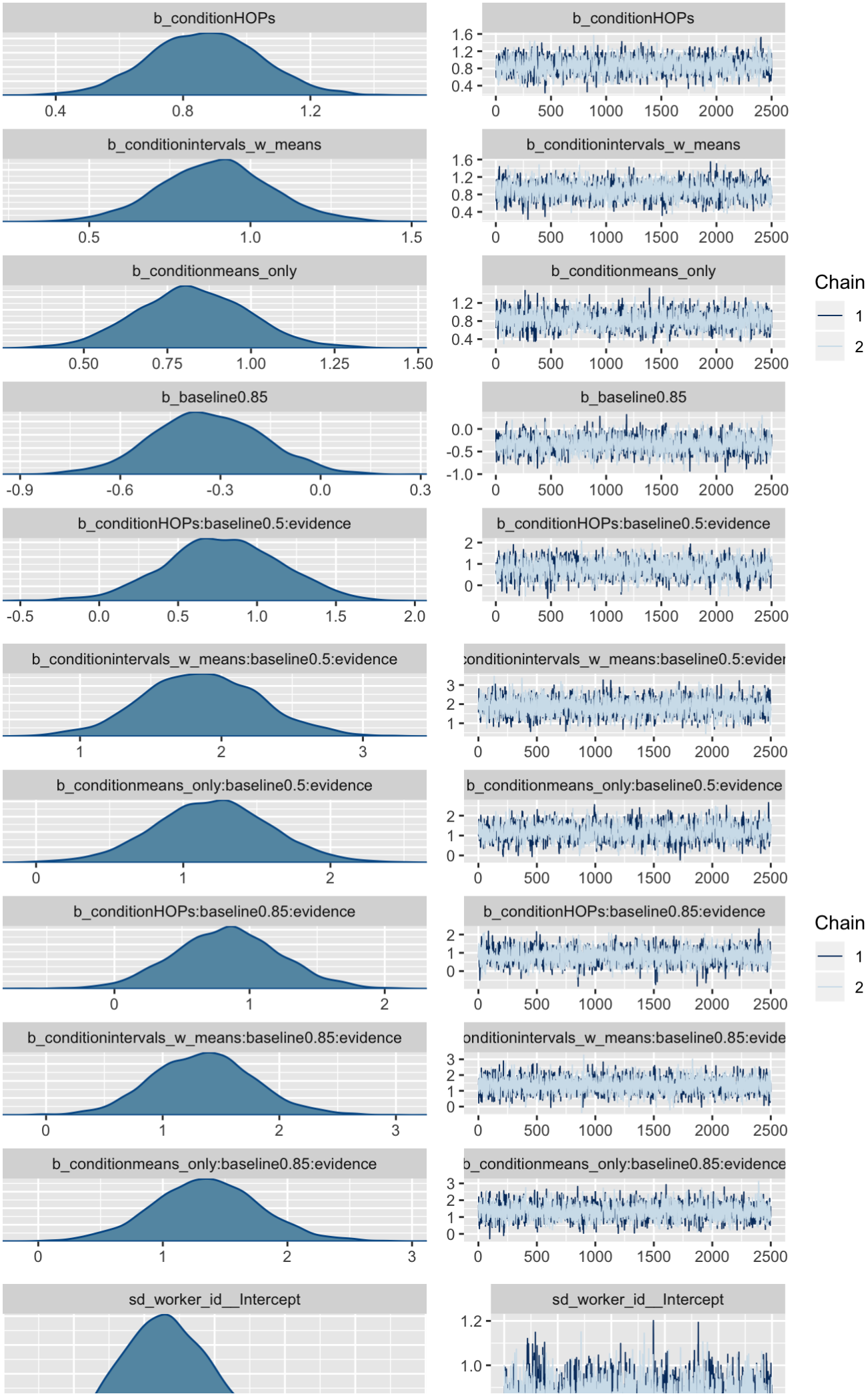
Similar to how we model the effects of visualization conditions on the location and slope of the psychometric function, we add predictors for baseline conditions. This gives us a three way interaction with evidence.

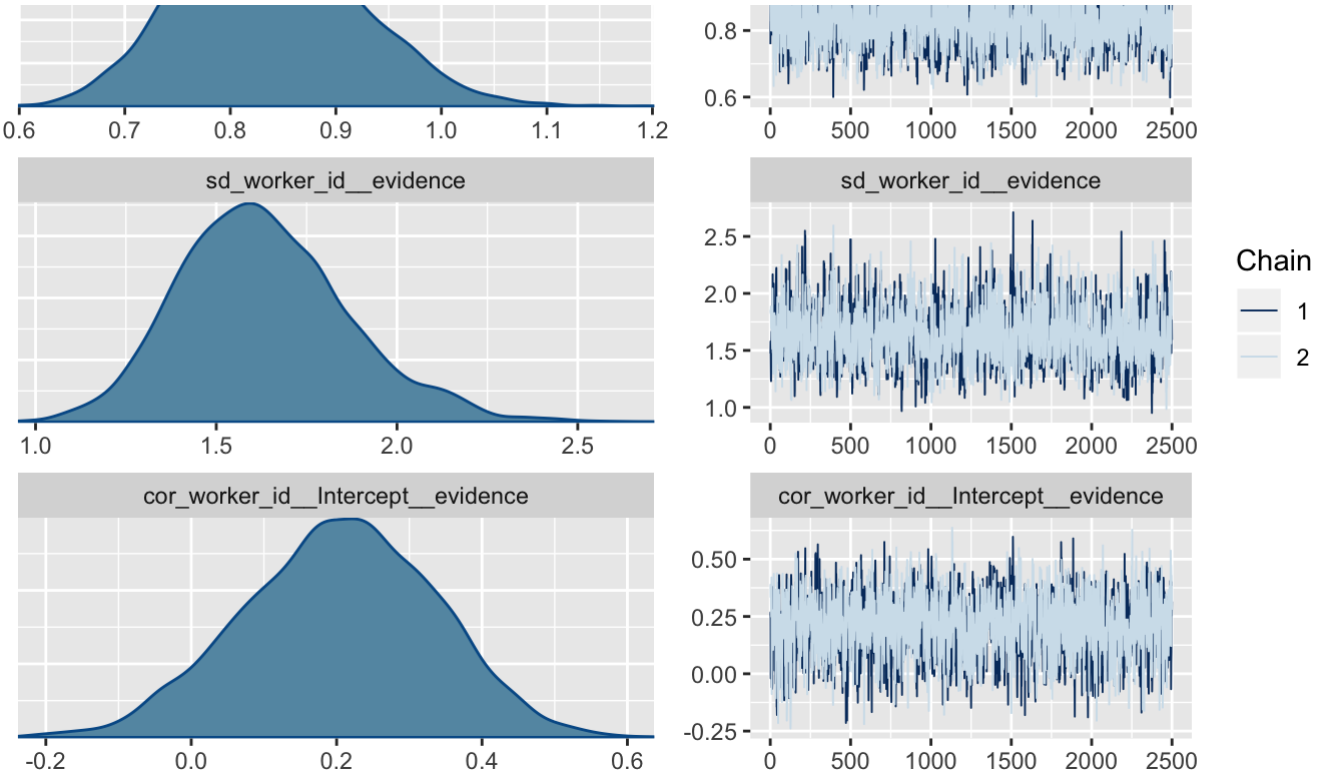
```
# linear model with probit link
m.vis.base.wrkr.probit <- brm(data = model_df, family = bernoulli(link = "probit"),
                              formula = bf(intervene ~ (1 + evidence|worker_id) + 0 +
condition + baseline + evidence:condition:baseline),
                              prior = c(prior(normal(0, 1), class = b)),
                              iter = 3000, warmup = 500, chains = 2, cores = 2,
                              file = "model-fits/probit_mdl_vis_base_wrkr")
```

Check diagnostics:

- Trace plots

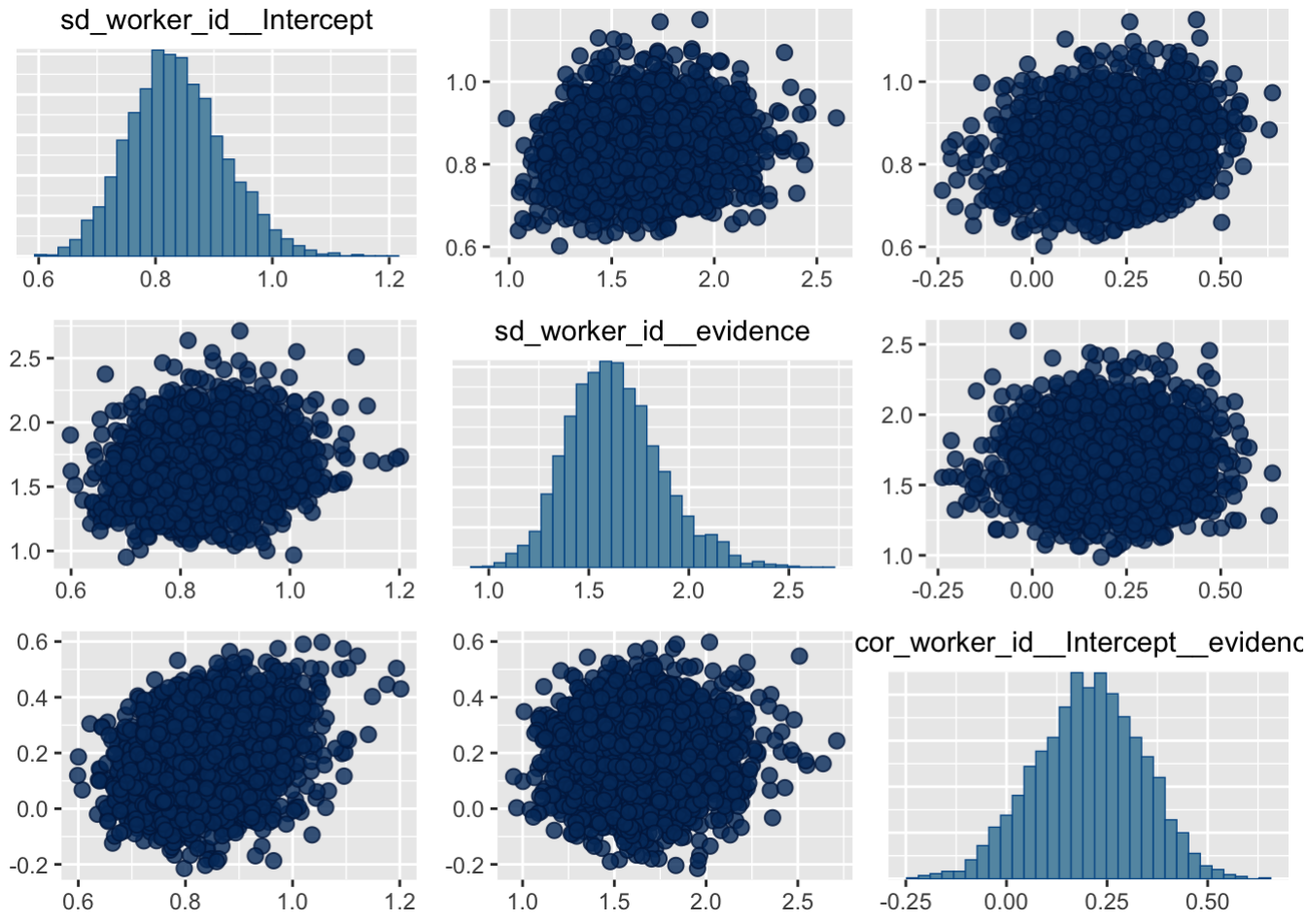
```
# trace plots
plot(m.vis.base.wrkr.probit)
```


b_conditionHOPs
b_conditionintervals_w_means
b_conditionmeans_only
b_baseline0.85
b_conditionHOPs:baseline0.5:evidence
b_conditionintervals_w_means:baseline0.5:evidence
b_conditionmeans_only:baseline0.5:evidence
b_conditionHOPs:baseline0.85:evidence
b_conditionintervals_w_means:baseline0.85:evidence
b_conditionmeans_only:baseline0.85:evidence
sd_worker_id__Intercept
Chain
1
2

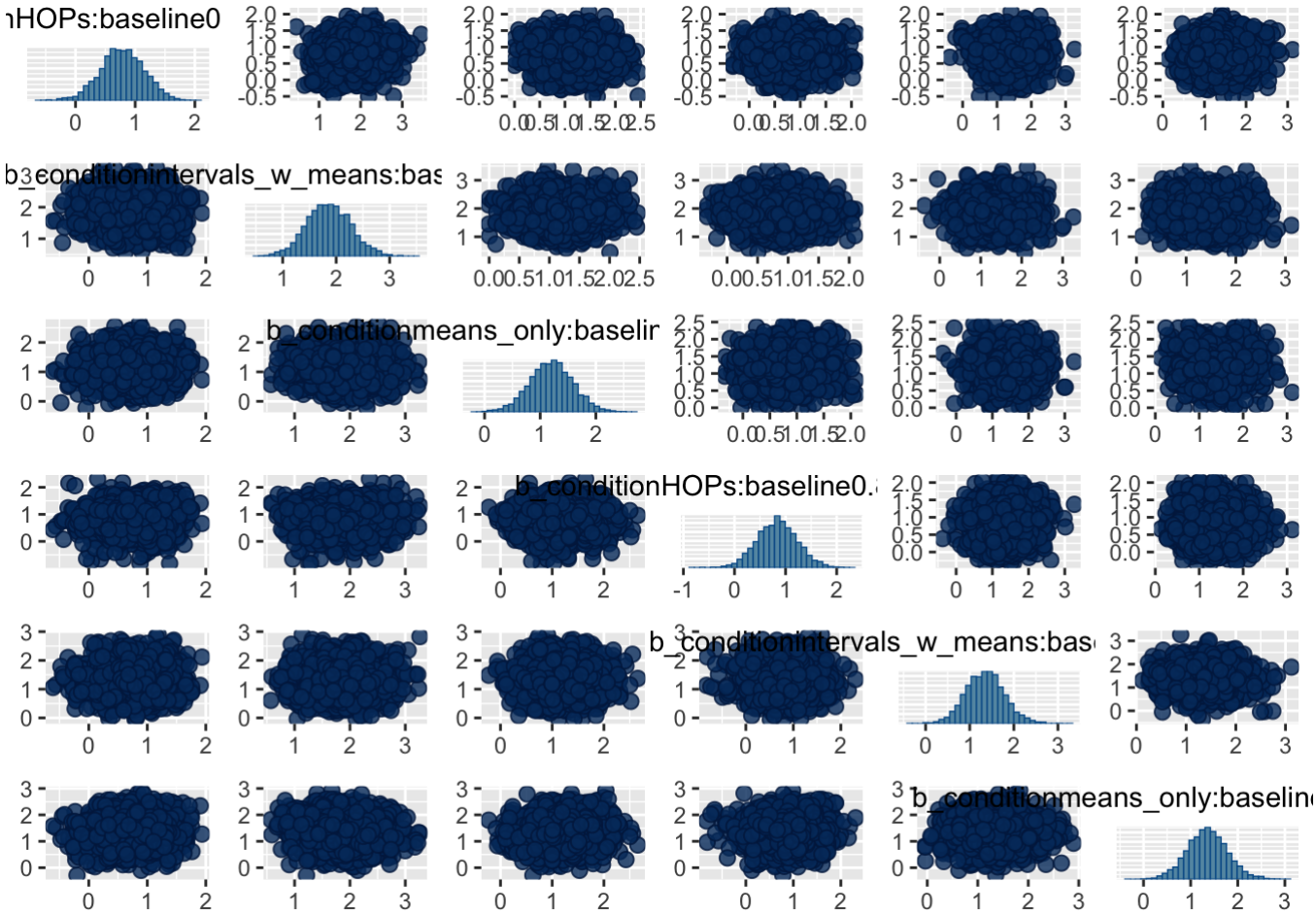


- Pairs plot

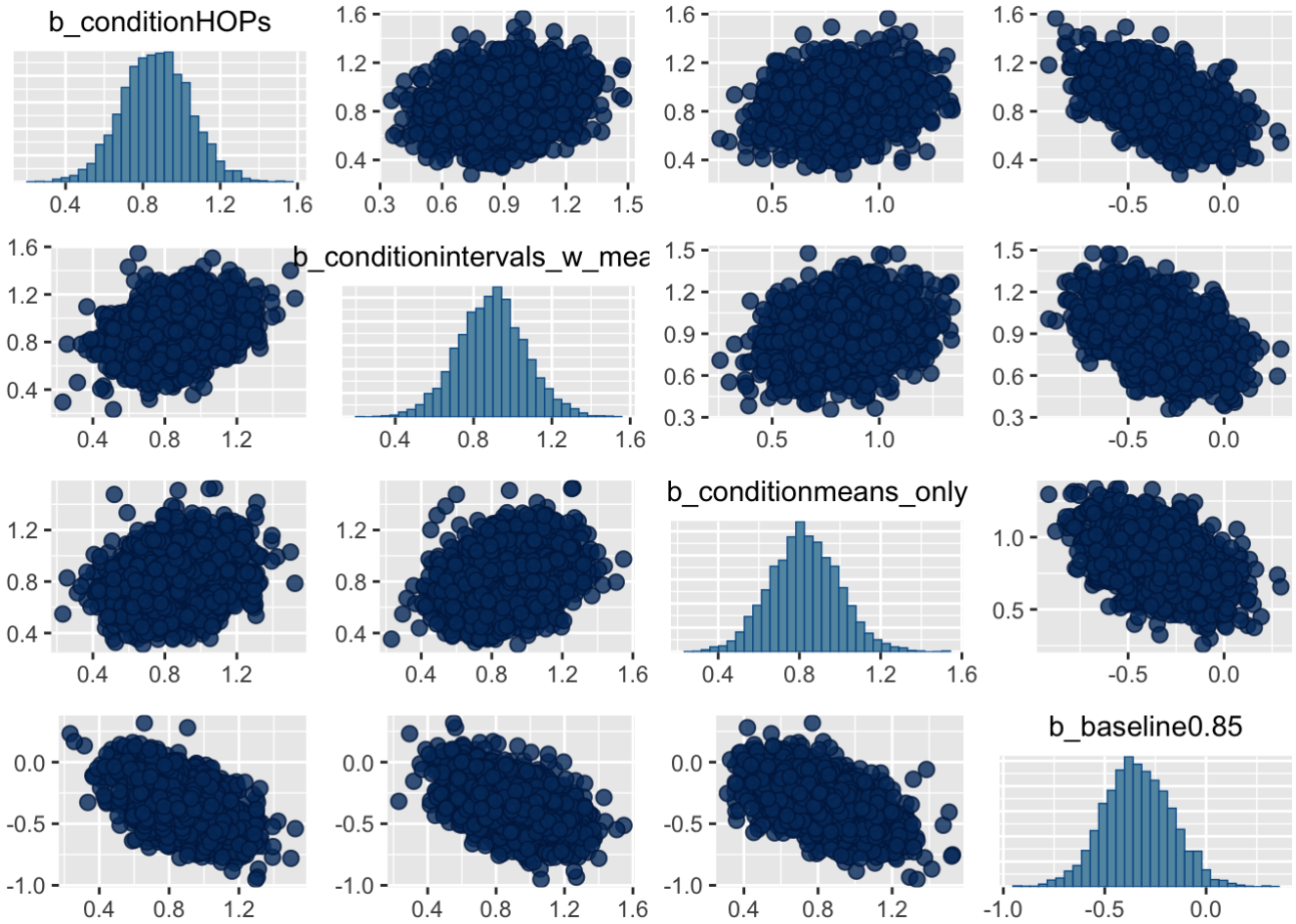
```
# pairs plot (too many things to view at once, so we've grouped them)
# hyperparameters
pairs(m.vis.base.wrkr.probit, pars = c("sd_worker_id__Intercept",
                                       "sd_worker_id__evidence",
                                       "cor_worker_id__Intercept__evidence"))
```



```
# pairs plot (too many things to view at once, so we've grouped them)
# slope effects
pairs(m.vis.base.wrkr.probit, pars = c("b_conditionHOPs:baseline0.5:evidence",
                                       "b_conditionintervals_w_means:baseline0.5:evid
ence",
                                       "b_conditionmeans_only:baseline0.5:evidence",
                                       "b_conditionHOPs:baseline0.85:evidence",
                                       "b_conditionintervals_w_means:baseline0.85:evi
dence",
                                       "b_conditionmeans_only:baseline0.85:evidence"
))
```




```
# pairs plot (too many things to view at once, so we've grouped them)
# intercept effects
pairs(m.vis.base.wrkr.probit, exact_match = TRUE, pars = c("b_conditionHOPs",
                                                           "b_conditionintervals_w_me
ans",
                                                           "b_conditionmeans_only",
                                                           "b_baseline0.85"))
```



- Summary

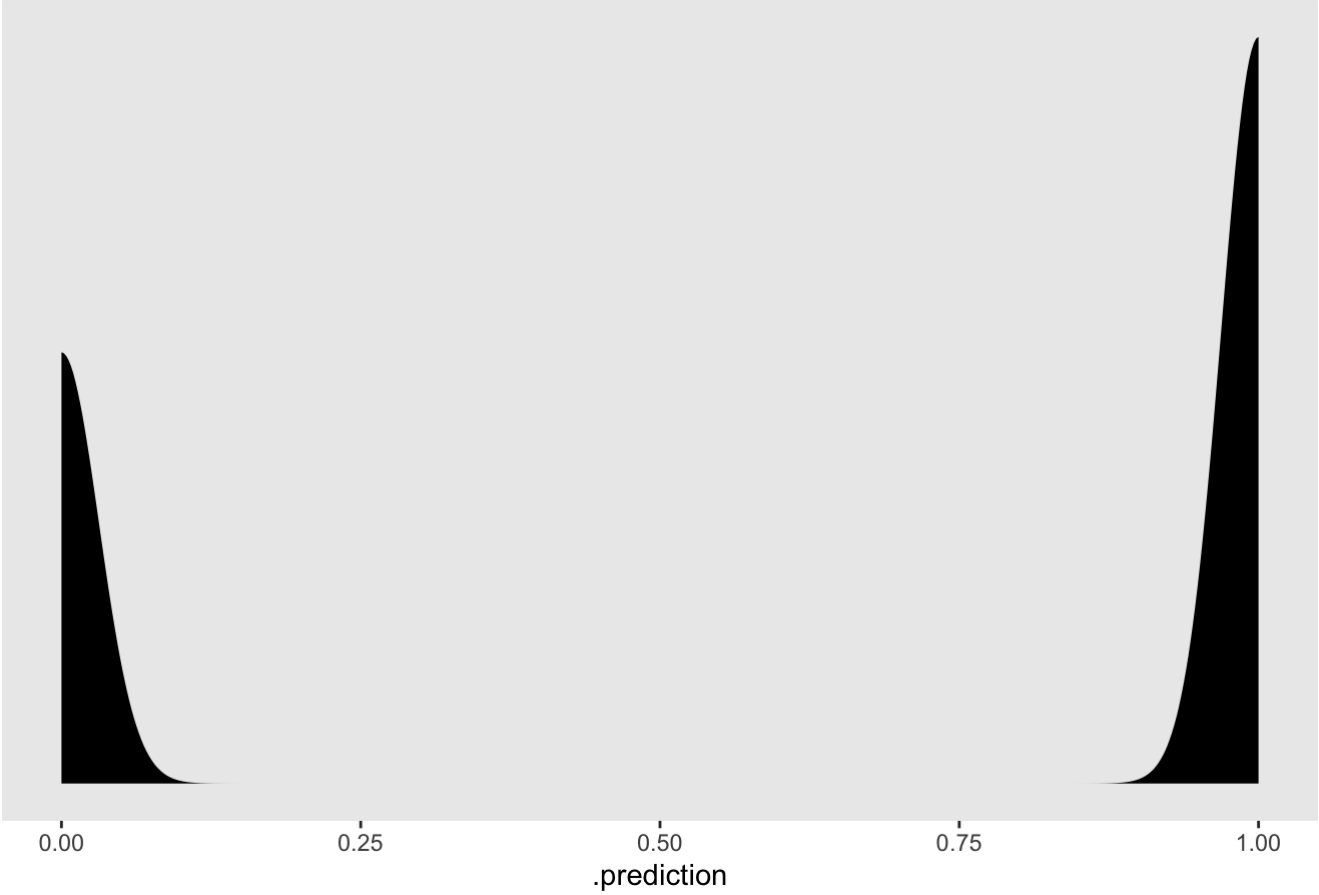
```
# model summary
print(m.vis.base.wrkr.probit)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ (1 + evidence | worker_id) + 0 + condition + baseline + evide
nce:condition:baseline
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Group-Level Effects:
## ~worker_id (Number of levels: 109)
##                         Estimate Est.Error l-95% CI u-95% CI Eff.Sample
## sd(Intercept)               0.84      0.08     0.69     1.01       1640
## sd(evidence)                1.63      0.24     1.22     2.15       1167
## cor(Intercept,evidence)     0.20      0.13    -0.06     0.45       1348
##                         Rhat
## sd(Intercept)           1.00
## sd(evidence)            1.00
## cor(Intercept,evidence) 1.00
##
## Population-Level Effects:
##                                                 Estimate Est.Error
## conditionHOPs                                       0.87      0.18
## conditionintervals_w_means                          0.89      0.17
## conditionmeans_only                                 0.83      0.17
## baseline0.85                                       -0.34      0.17
## conditionHOPs:baseline0.5:evidence                  0.78      0.37
## conditionintervals_w_means:baseline0.5:evidence     1.86      0.41
## conditionmeans_only:baseline0.5:evidence            1.20      0.39
## conditionHOPs:baseline0.85:evidence                 0.83      0.40
## conditionintervals_w_means:baseline0.85:evidence    1.35      0.44
## conditionmeans_only:baseline0.85:evidence           1.35      0.43
##                                                 l-95% CI u-95% CI
## conditionHOPs                                       0.53     1.21
## conditionintervals_w_means                          0.55     1.24
## conditionmeans_only                                 0.50     1.17
## baseline0.85                                       -0.66    -0.01
## conditionHOPs:baseline0.5:evidence                  0.05     1.49
## conditionintervals_w_means:baseline0.5:evidence     1.05     2.70
## conditionmeans_only:baseline0.5:evidence            0.42     1.97
## conditionHOPs:baseline0.85:evidence                 0.07     1.63
## conditionintervals_w_means:baseline0.85:evidence    0.52     2.24
## conditionmeans_only:baseline0.85:evidence           0.51     2.21
##                                                 Eff.Sample Rhat
## conditionHOPs                                         1503 1.00
## conditionintervals_w_means                            1774 1.00
## conditionmeans_only                                   1504 1.00
## baseline0.85                                          1697 1.00
## conditionHOPs:baseline0.5:evidence                    1284 1.00
## conditionintervals_w_means:baseline0.5:evidence       1807 1.00
## conditionmeans_only:baseline0.5:evidence              1563 1.00
## conditionHOPs:baseline0.85:evidence                   2058 1.00
## conditionintervals_w_means:baseline0.85:evidence      2621 1.00
## conditionmeans_only:baseline0.85:evidence             2485 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  group_by(worker_id, condition, evidence) %>%
  add_predicted_draws(m.vis.base.wrkr.probit, seed = 1234, n = 200) %>%
  ggplot(aes(x = .prediction)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

Posterior predictive distribution for intervention

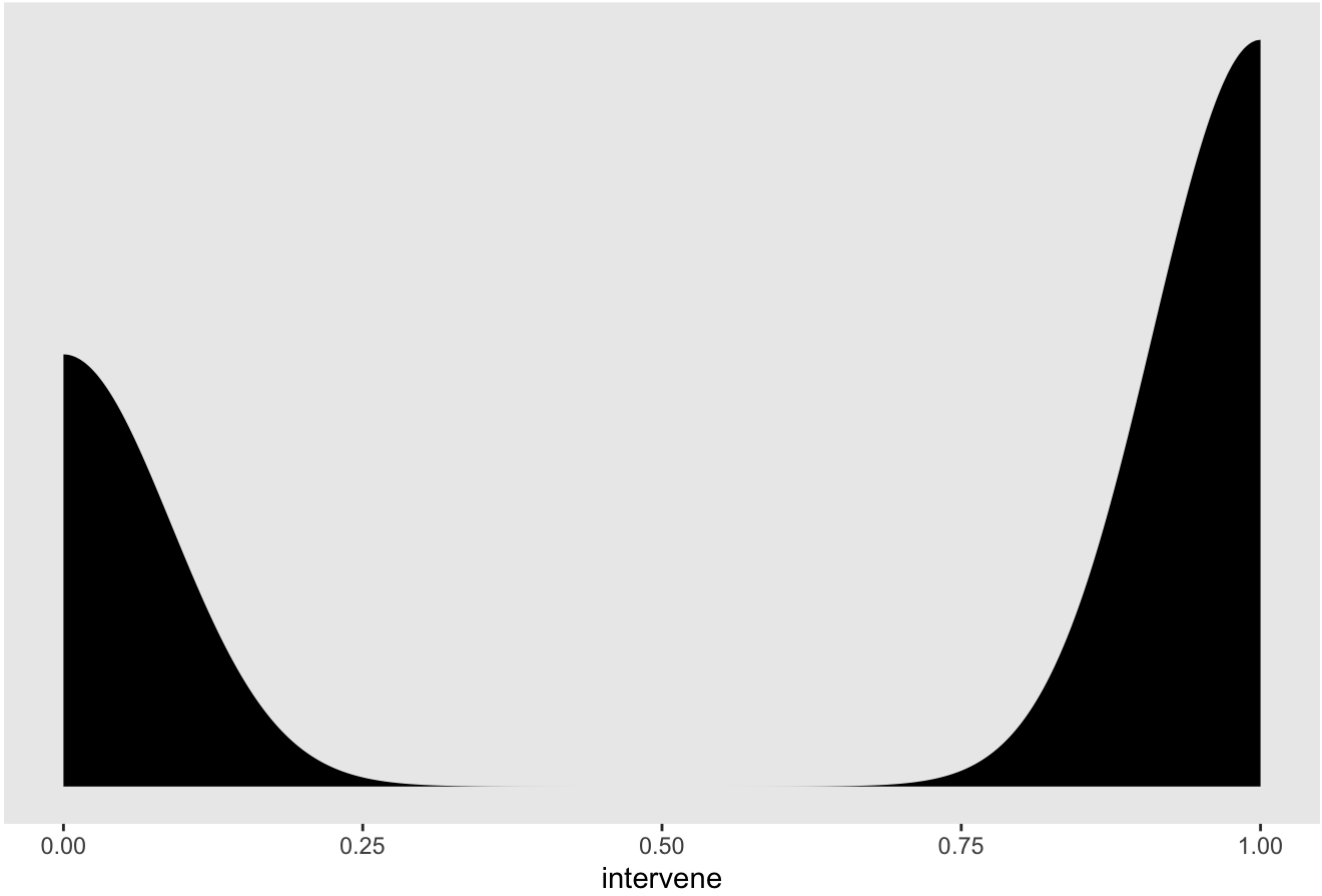


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

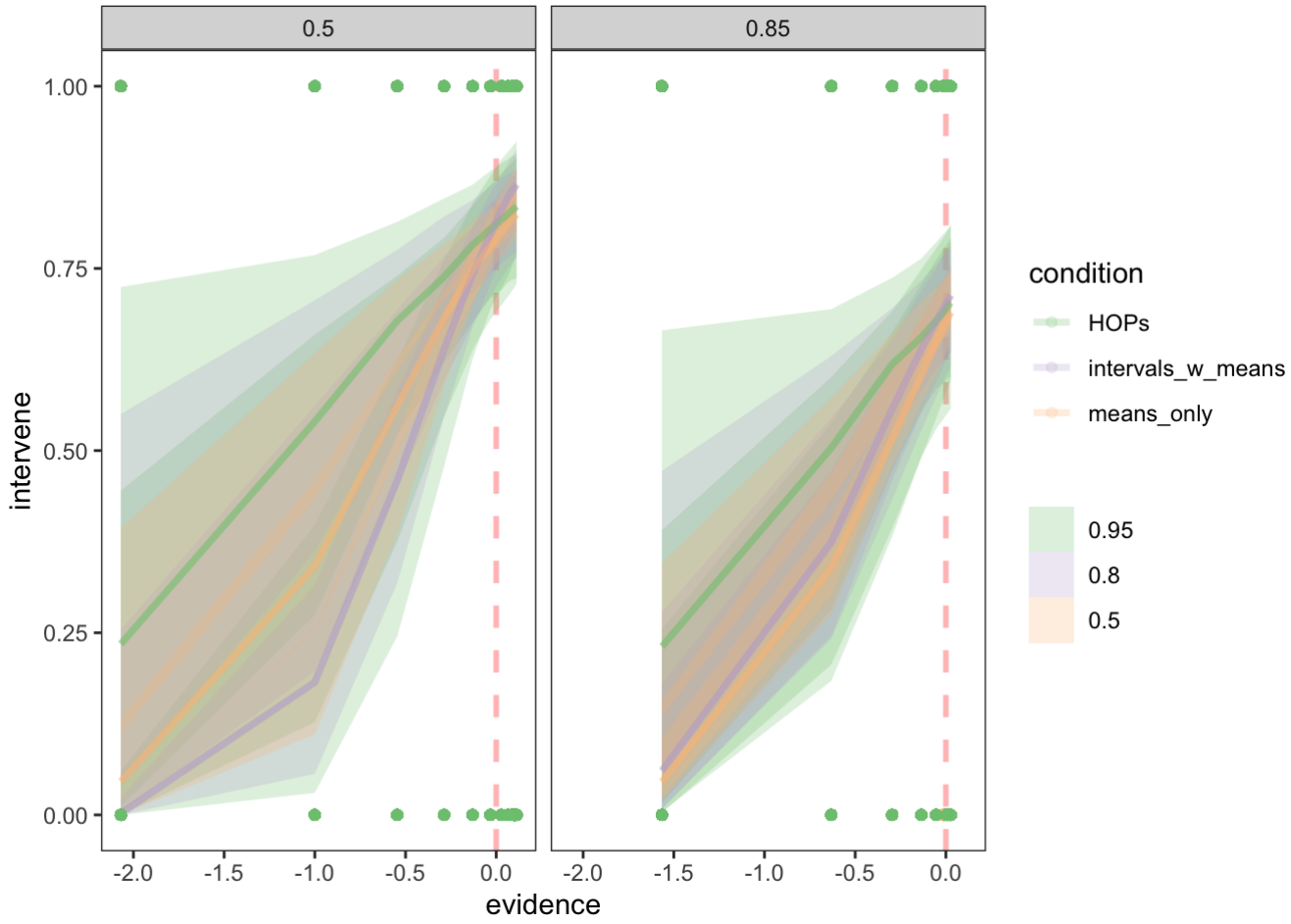
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric function in each condition for the average observer.

```
model_df %>%
  group_by(evidence, condition, baseline, worker_id) %>%
  add_fitted_draws(m.vis.base.wrkr.probit, value = "pf", re_formula = NA, n = 200) %
>%
  ggplot(aes(x = evidence, y = intervene, color = condition)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15) +
  scale_fill_brewer(type = "qual", palette = 1) +
  scale_color_brewer(type = "qual", palette = 1) +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank()) +
  facet_grid(. ~ baseline)
```



Psychometric function fits look qualitatively similar across baseline conditions, however, performance is closer across conditions in the high baseline condition. While it looks like we are hardly sampling above the utitlity optimal decision threshold in the baseline == 0.85 condition, half of the trials we sample are above an evidence level of 0. However, the skew in sampling on the evidence scale might not be ideal for fitting psychometric functions.

```
model_df %>%
  mutate(above_threshold = evidence > 0) %>%
  group_by(baseline, condition, frame) %>%
  summarise(
    proportion_above_threshold = sum(above_threshold) / n()
  )
```

```
## # A tibble: 12 x 4
## # Groups:   baseline, condition [6]
##    baseline condition         frame proportion_above_threshold
##    <fct>    <chr>             <fct>                       <dbl>
##  1 0.5      HOPs              gain                          0.5
##  2 0.5      HOPs              loss                          0.5
##  3 0.5      intervals_w_means gain                          0.5
##  4 0.5      intervals_w_means loss                          0.5
##  5 0.5      means_only        gain                          0.5
##  6 0.5      means_only        loss                          0.5
##  7 0.85     HOPs              gain                          0.5
##  8 0.85     HOPs              loss                          0.5
##  9 0.85     intervals_w_means gain                          0.5
## 10 0.85     intervals_w_means loss                          0.5
## 11 0.85     means_only        gain                          0.5
## 12 0.85     means_only        loss                          0.5
```

Use Predictors for Problem Framing Instead of Predictors for Baseline Condition

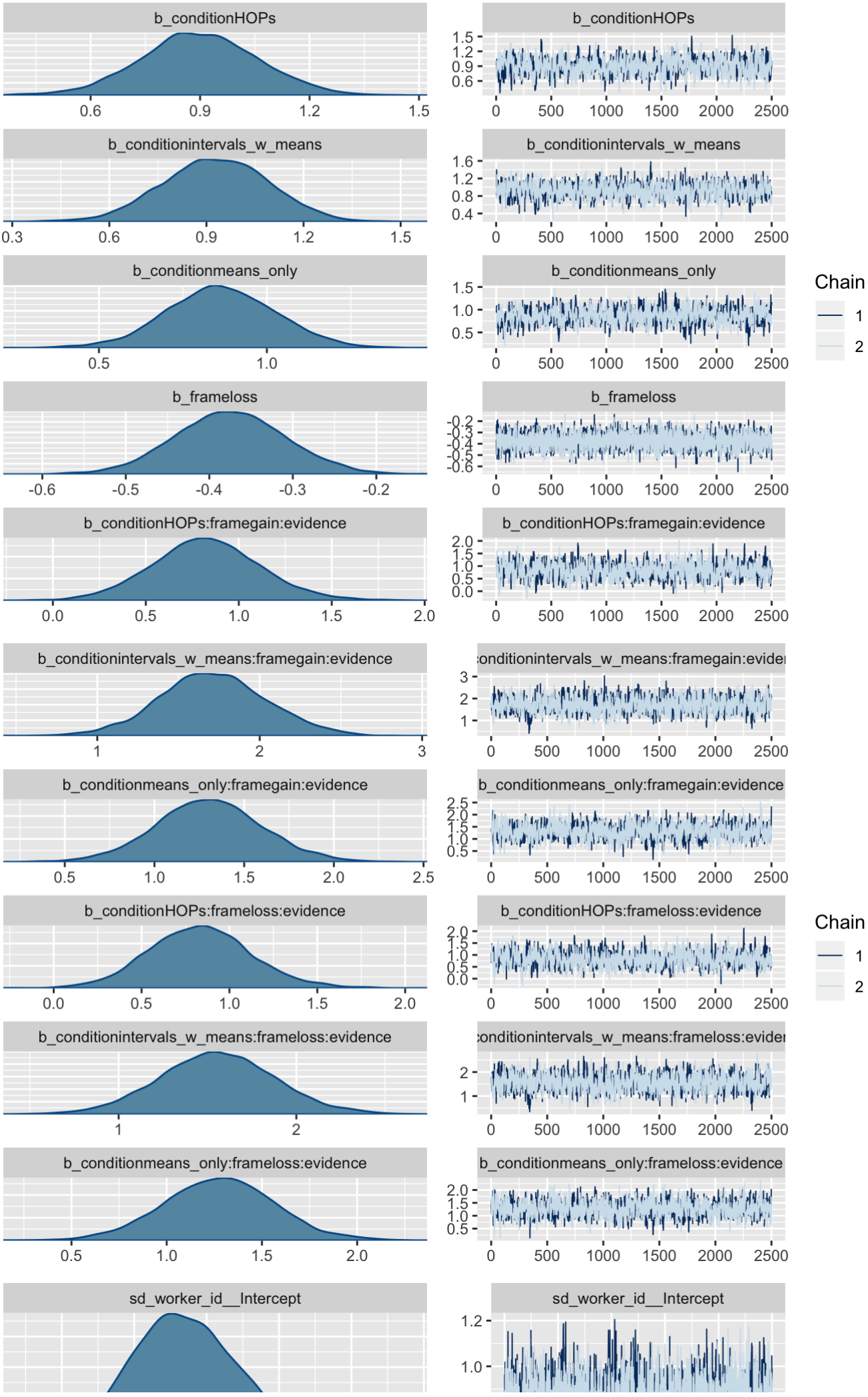
Let's take the same approach to modeling the effect of problem framing instead of baseline condition. After this we'll put together a model with both baseline and framing parameters, and we'll compare all of our models thus far.

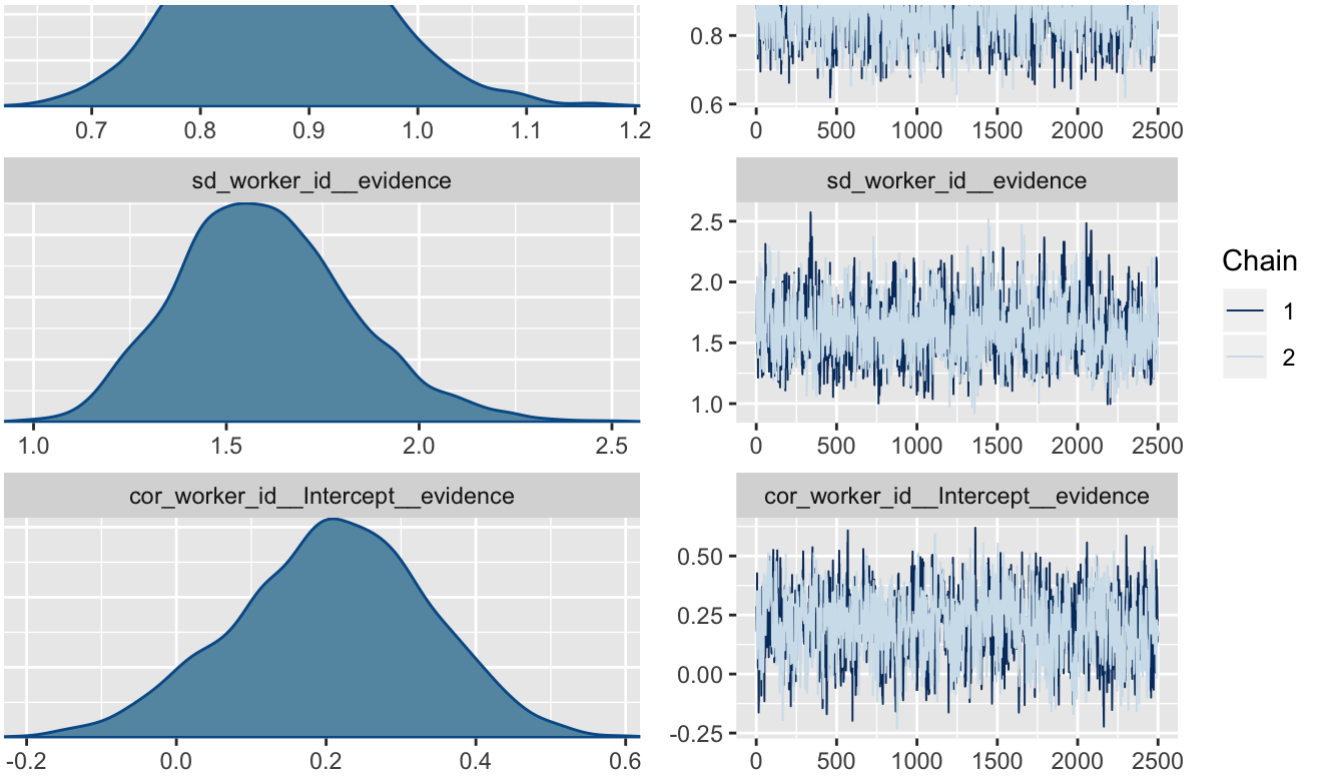
```
# linear model with probit link
m.vis.frame.wrkr.probit <- brm(data = model_df, family = bernoulli(link = "probit"),
                               formula = bf(intervene ~ (1 + evidence|worker_id) + 0 +
condition + frame + evidence:condition:frame),
                               prior = c(prior(normal(0, 1), class = b)),
                               iter = 3000, warmup = 500, chains = 2, cores = 2,
                               file = "model-fits/probit_mdl_vis_frame_wrkr")
```

Check diagnostics:

- Trace plots

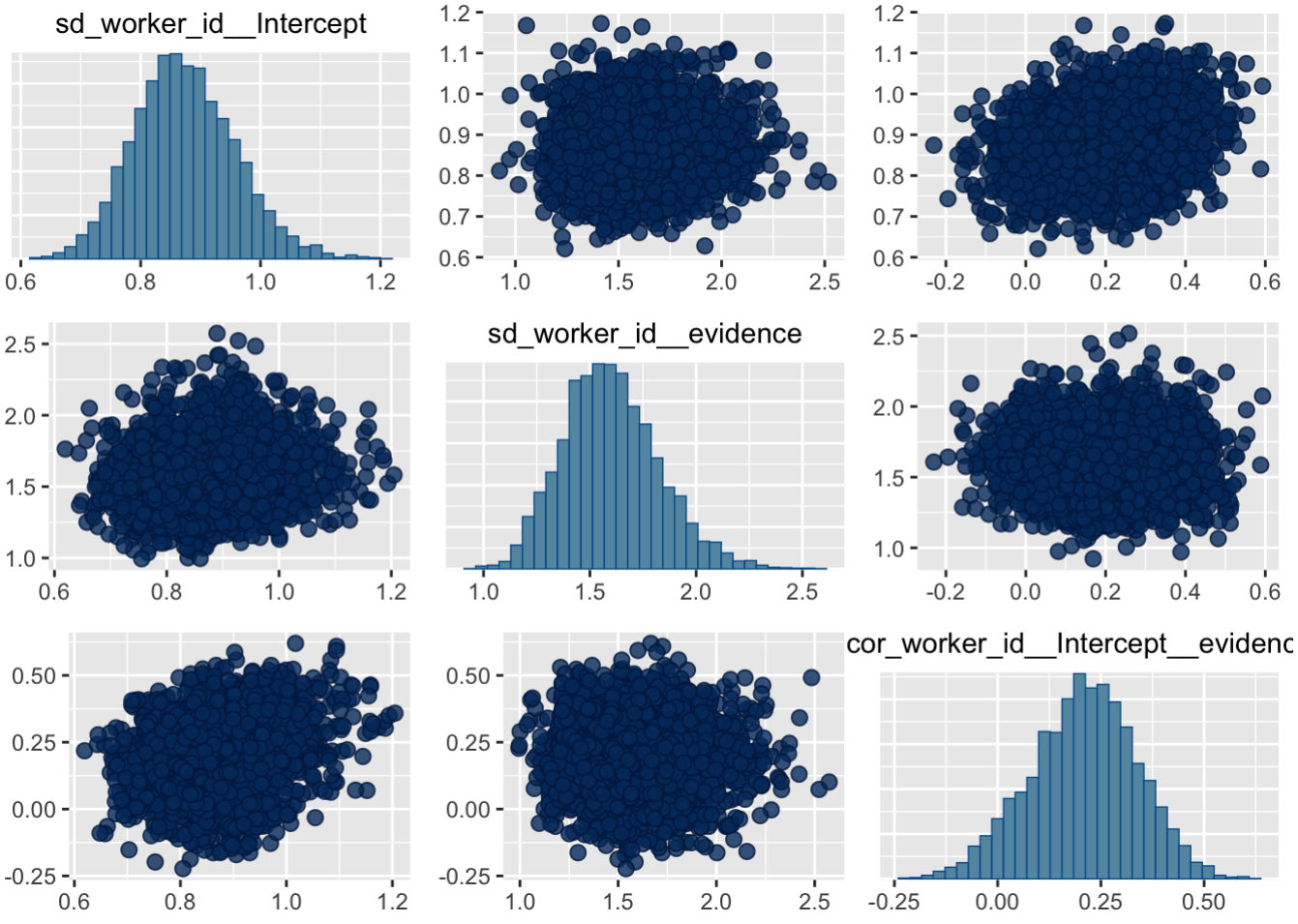
```
# trace plots
plot(m.vis.frame.wrkr.probit)
```


b_conditionHOPs
b_conditionintervals_w_means
b_conditionmeans_only
b_frameloss
b_conditionHOPs:framegain:evidence
b_conditionintervals_w_means:framegain:evidence
b_conditionmeans_only:framegain:evidence
b_conditionHOPs:frameloss:evidence
b_conditionintervals_w_means:frameloss:evidence
b_conditionmeans_only:frameloss:evidence
sd_worker_id__Intercept
Chain
1
2

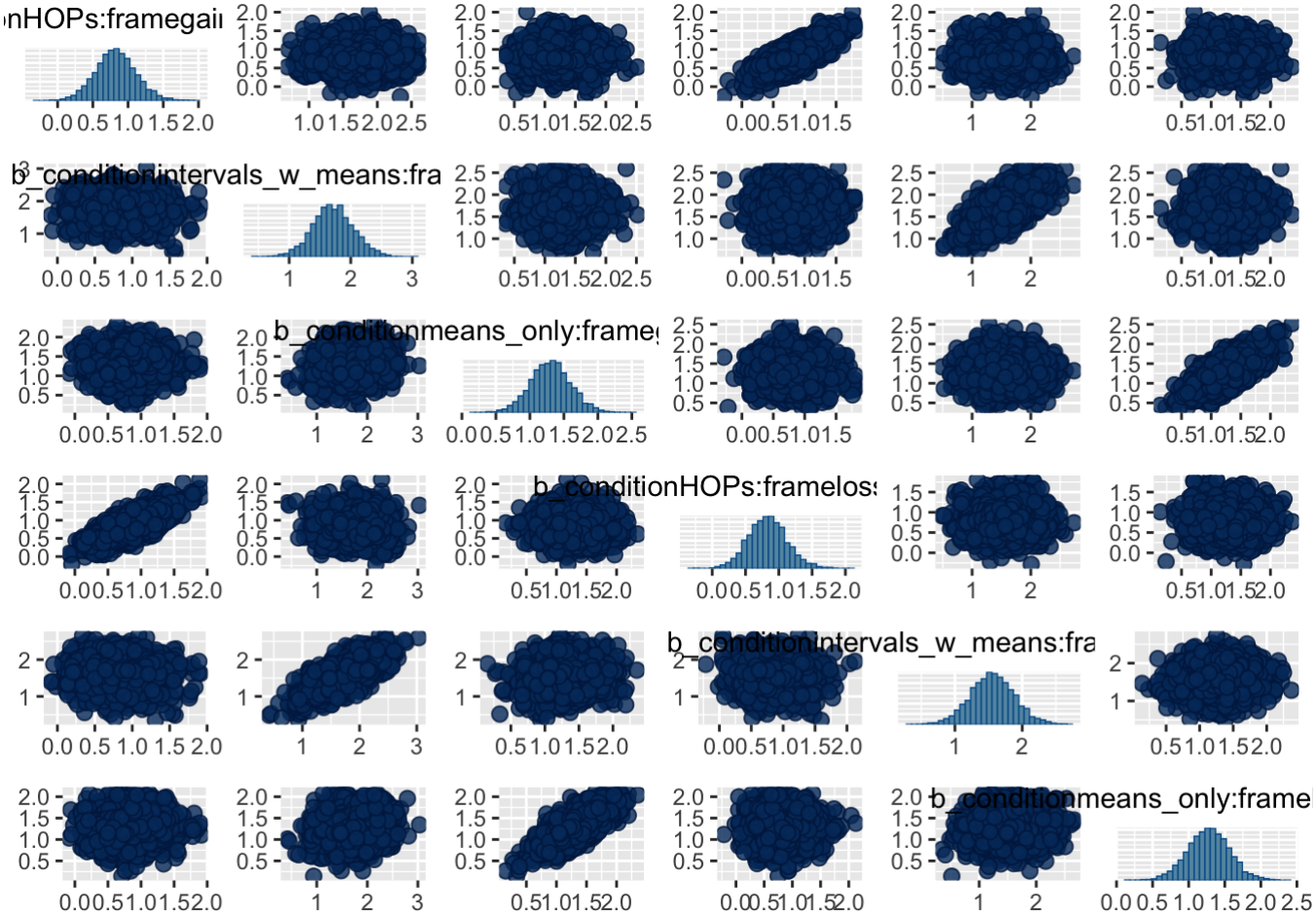


- Pairs plot

```
# pairs plot (too many things to view at once, so we've grouped them)
# hyperparameters
pairs(m.vis.frame.wrkr.probit, pars = c("sd_worker_id__Intercept",
                                        "sd_worker_id__evidence",
                                        "cor_worker_id__Intercept__evidence"))
```

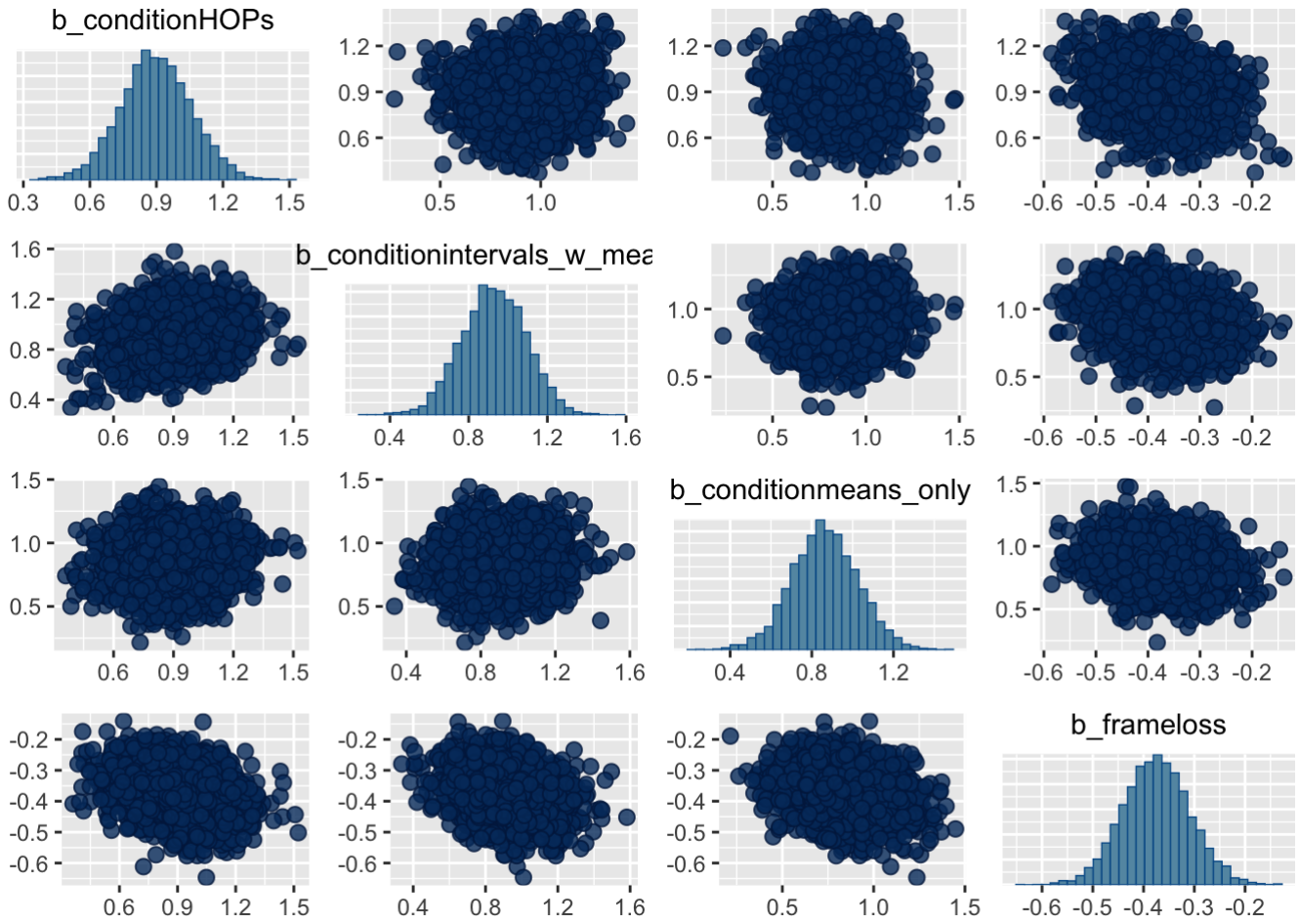


```
# pairs plot (too many things to view at once, so we've grouped them)
# slope effects
pairs(m.vis.frame.wrkr.probit, pars = c("b_conditionHOPs:framegain:evidence",
                                        "b_conditionintervals_w_means:framegain:eviden
ce",
                                        "b_conditionmeans_only:framegain:evidence",
                                        "b_conditionHOPs:frameloss:evidence",
                                        "b_conditionintervals_w_means:frameloss:eviden
ce",
                                        "b_conditionmeans_only:frameloss:evidence"))
```



The level of correlation among slopes in the gain frame is somewhat concerning.

```
# pairs plot (too many things to view at once, so we've grouped them)
# intercept effects
pairs(m.vis.frame.wrkr.probit, exact_match = TRUE, pars = c("b_conditionHOPs",
                                                            "b_conditionintervals_w_me
ans",
                                                            "b_conditionmeans_only",
                                                            "b_frameloss"))
```



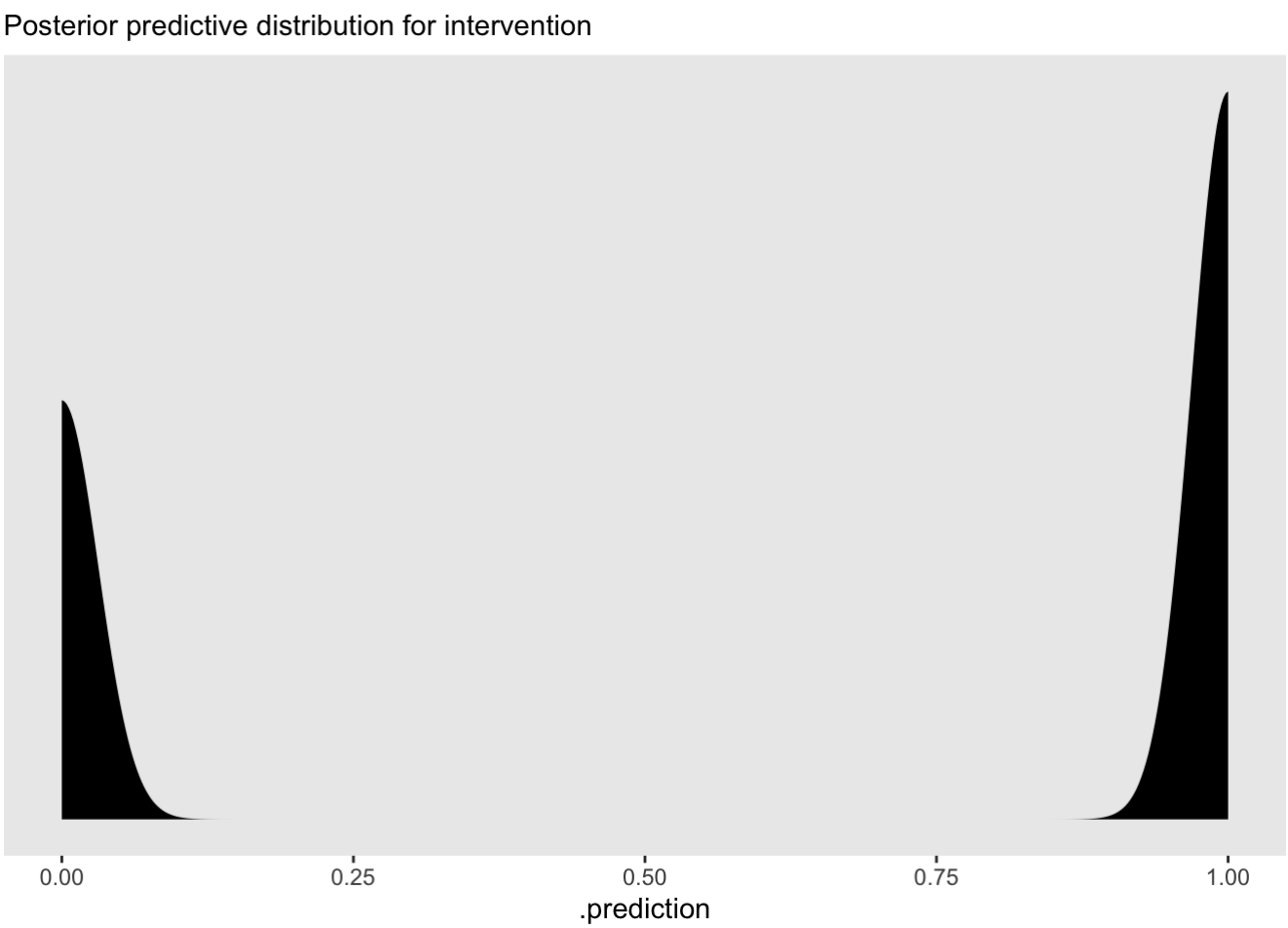
- Summary

```
# model summary
print(m.vis.frame.wrkr.probit)
```

```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ (1 + evidence | worker_id) + 0 + condition + frame + evidenc
e:condition:frame
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Group-Level Effects:
## ~worker_id (Number of levels: 109)
##                         Estimate Est.Error l-95% CI u-95% CI Eff.Sample
## sd(Intercept)               0.87      0.08     0.72     1.05       1196
## sd(evidence)                1.60      0.23     1.20     2.09        636
## cor(Intercept,evidence)     0.21      0.13    -0.06     0.45        750
##                         Rhat
## sd(Intercept)           1.00
## sd(evidence)            1.00
## cor(Intercept,evidence) 1.00
##
## Population-Level Effects:
##                                             Estimate Est.Error l-95% CI
## conditionHOPs                                   0.90      0.16     0.58
## conditionintervals_w_means                      0.93      0.16     0.61
## conditionmeans_only                             0.86      0.16     0.54
## frameloss                                      -0.37      0.07    -0.50
## conditionHOPs:framegain:evidence                0.83      0.29     0.27
## conditionintervals_w_means:framegain:evidence   1.71      0.33     1.08
## conditionmeans_only:framegain:evidence          1.30      0.30     0.72
## conditionHOPs:frameloss:evidence                0.83      0.29     0.28
## conditionintervals_w_means:frameloss:evidence   1.56      0.33     0.91
## conditionmeans_only:frameloss:evidence          1.28      0.29     0.69
##                                             u-95% CI Eff.Sample Rhat
## conditionHOPs                                   1.21        646 1.00
## conditionintervals_w_means                      1.24        921 1.00
## conditionmeans_only                             1.18        653 1.00
## frameloss                                      -0.24       4019 1.00
## conditionHOPs:framegain:evidence                1.41        771 1.00
## conditionintervals_w_means:framegain:evidence   2.37        904 1.00
## conditionmeans_only:framegain:evidence          1.90        940 1.00
## conditionHOPs:frameloss:evidence                1.43        757 1.00
## conditionintervals_w_means:frameloss:evidence   2.21        942 1.00
## conditionmeans_only:frameloss:evidence          1.86        995 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  group_by(worker_id, condition, evidence) %>%
  add_predicted_draws(m.vis.frame.wrkr.probit, seed = 1234, n = 200) %>%
  ggplot(aes(x = .prediction)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

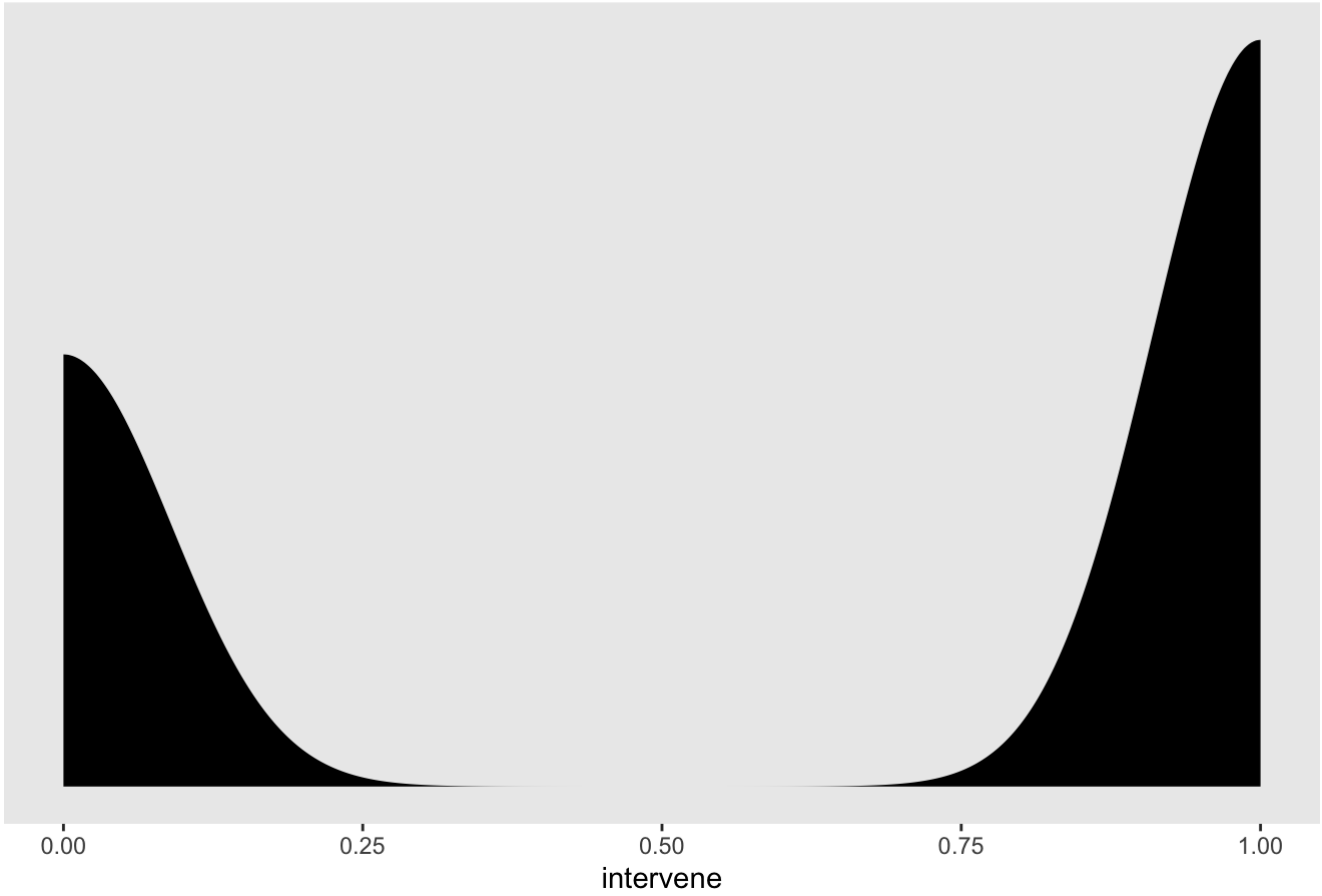


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

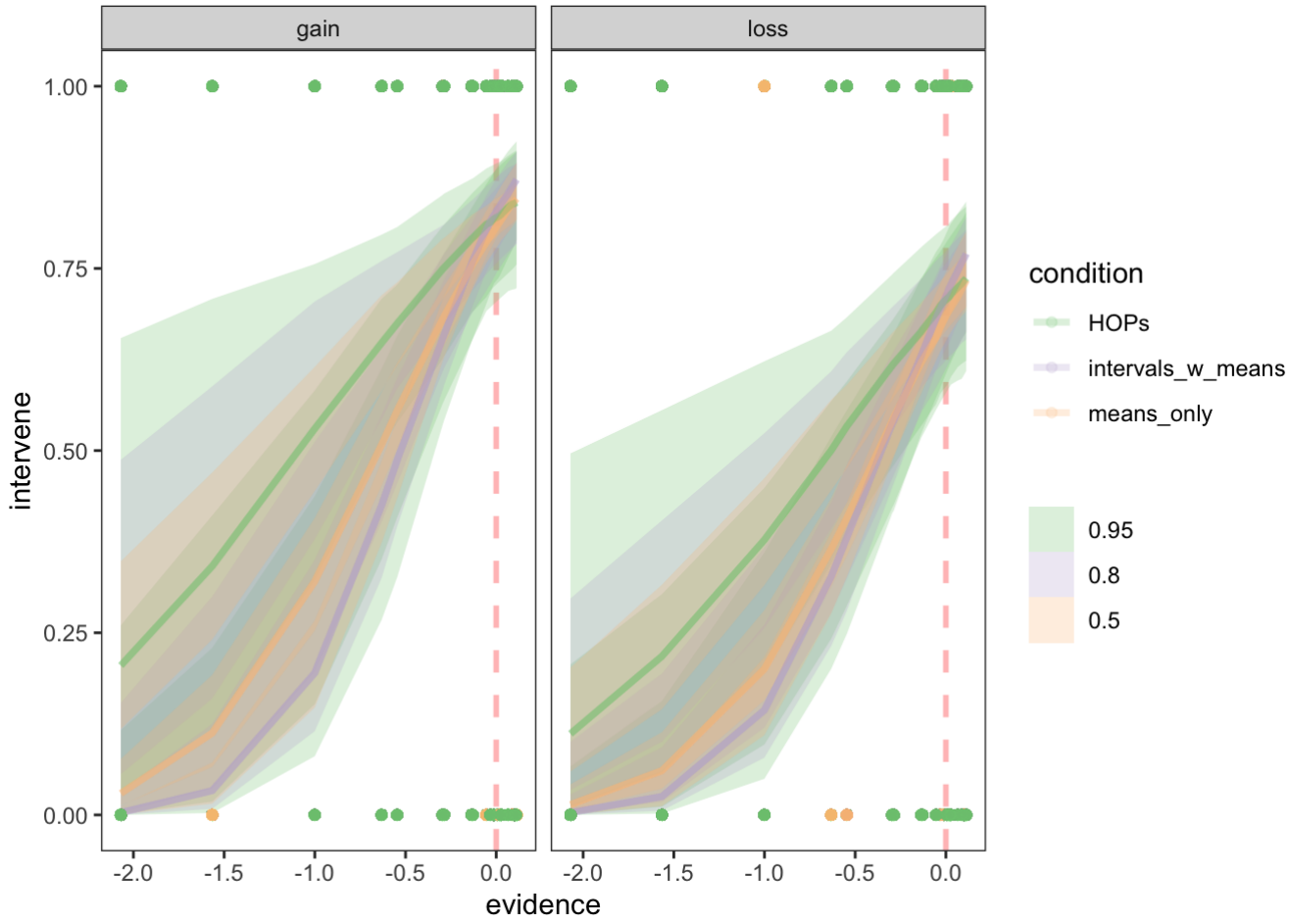
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric functions.

```
model_df %>%
  group_by(evidence, condition, frame, worker_id) %>%
  add_fitted_draws(m.vis.frame.wrkr.probit, value = "pf", re_formula = NA, n = 200) %
>%
  ggplot(aes(x = evidence, y = intervene, color = condition)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15) +
  scale_fill_brewer(type = "qual", palette = 1) +
  scale_color_brewer(type = "qual", palette = 1) +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank()) +
  facet_grid(. ~ frame)
```



Here we see a slight difference in psychometric functions for visualization conditions across problem frames, with more convergence of performance in the loss frame.

Use Predictors for Visualization, Baseline, and Problem Framing

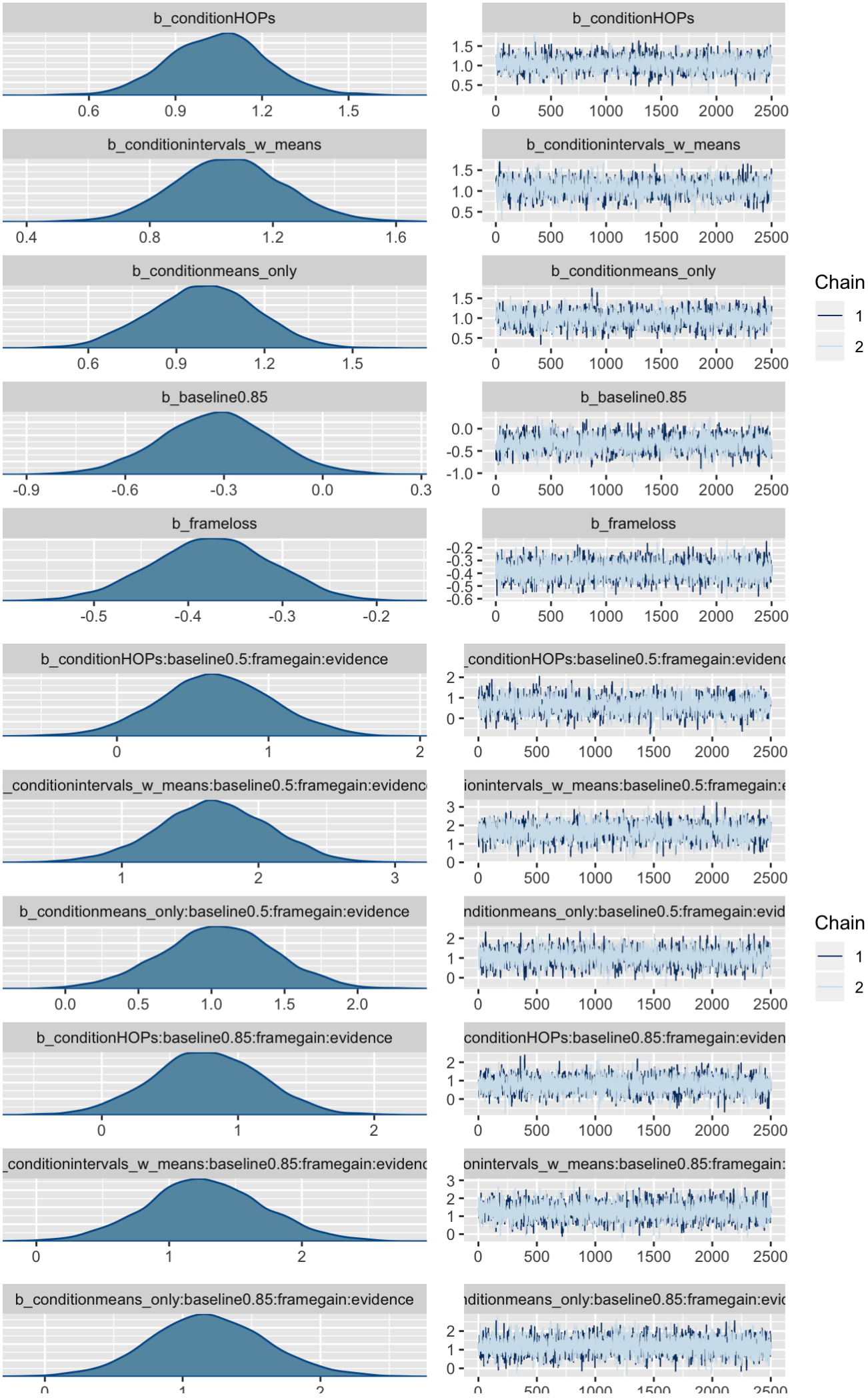
Let's put everything we've done so far into one model (with a four way interaction!).

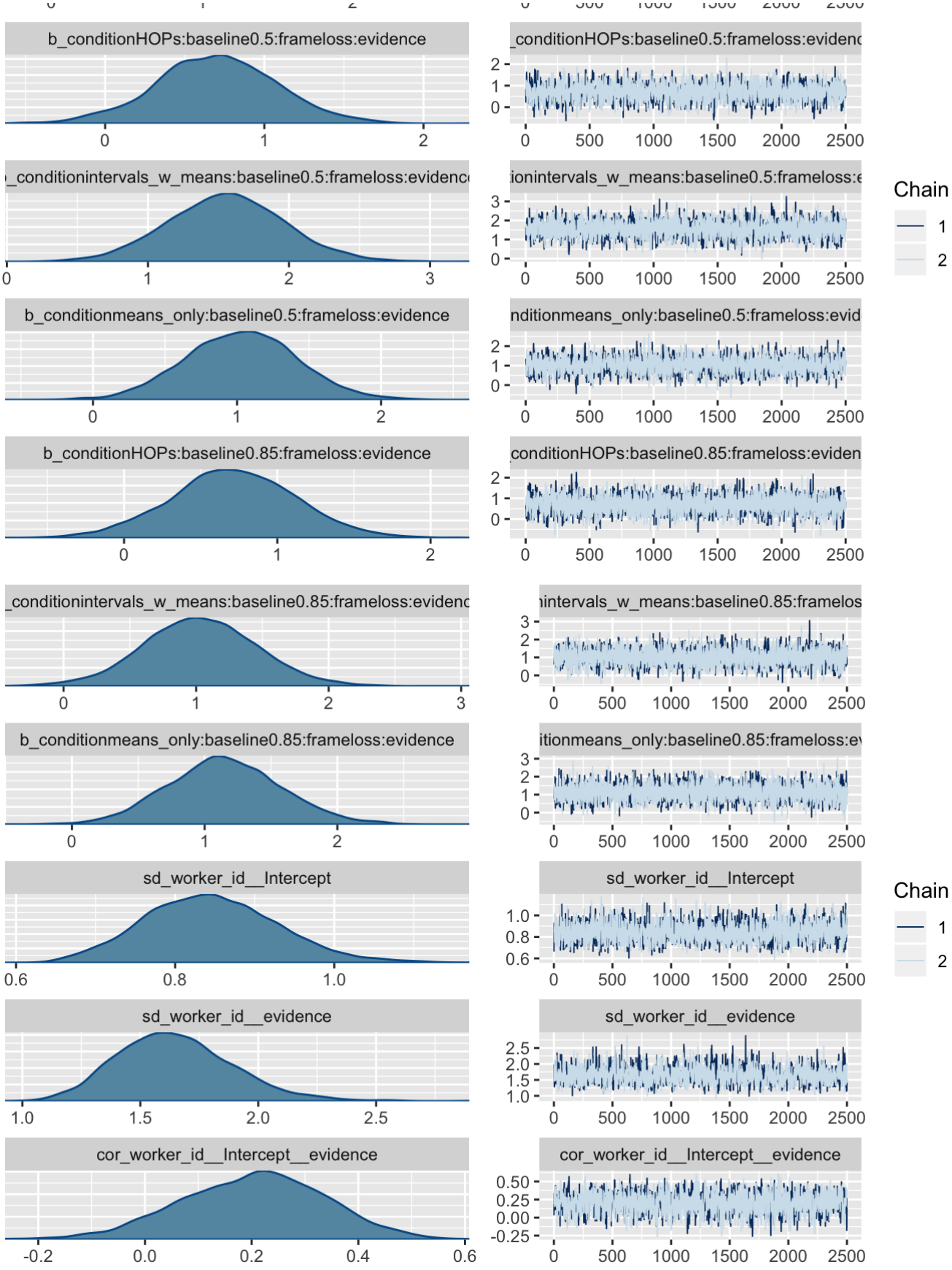
```
# linear model with probit link
m.vis.base.frame.wrkr.probit <- brm(data = model_df, family = bernoulli(link = "probi
t"),
                                    formula = bf(intervene ~ (1 + evidence|worker_id) + 0 +
condition + baseline + frame + evidence:condition:baseline:frame),
                                    prior = c(prior(normal(0, 1), class = b)),
                                    iter = 3000, warmup = 500, chains = 2, cores = 2,
                                    file = "model-fits/probit_mdl_vis_base_frame_wrkr")
```

Check diagnostics:

- Trace plots

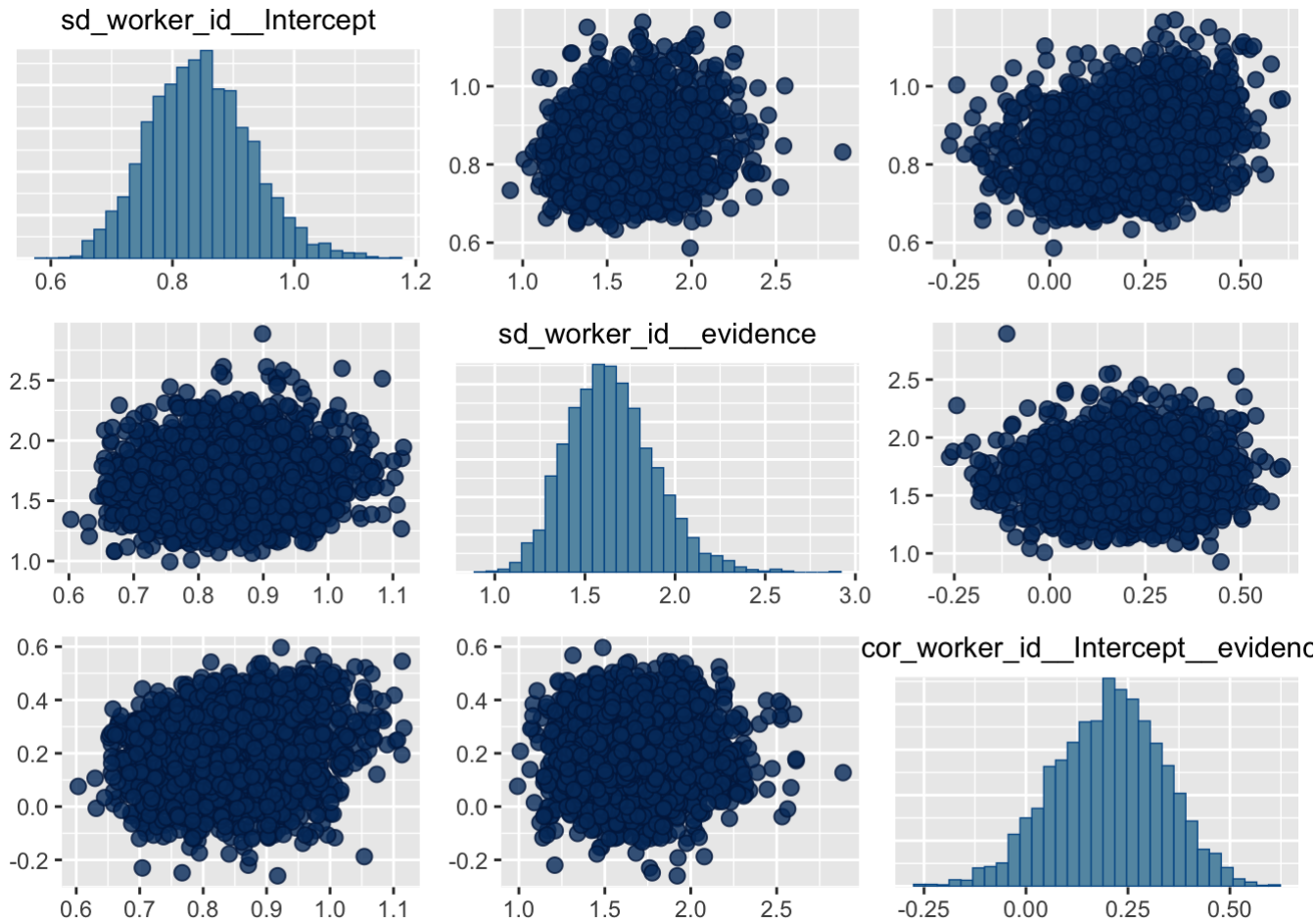
```
# trace plots
plot(m.vis.base.frame.wrkr.probit)
```


b_conditionHOPs
b_conditionintervals_w_means
b_conditionmeans_only
b_baseline0.85
b_frameloss
b_conditionHOPs:baseline0.5:framegain:evidence
_conditionintervals_w_means:baseline0.5:framegain:evidenc
b_conditionmeans_only:baseline0.5:framegain:evidence
b_conditionHOPs:baseline0.85:framegain:evidence
_conditionintervals_w_means:baseline0.85:framegain:evidenc
b_conditionmeans_only:baseline0.85:framegain:evidence
Chain
1
2

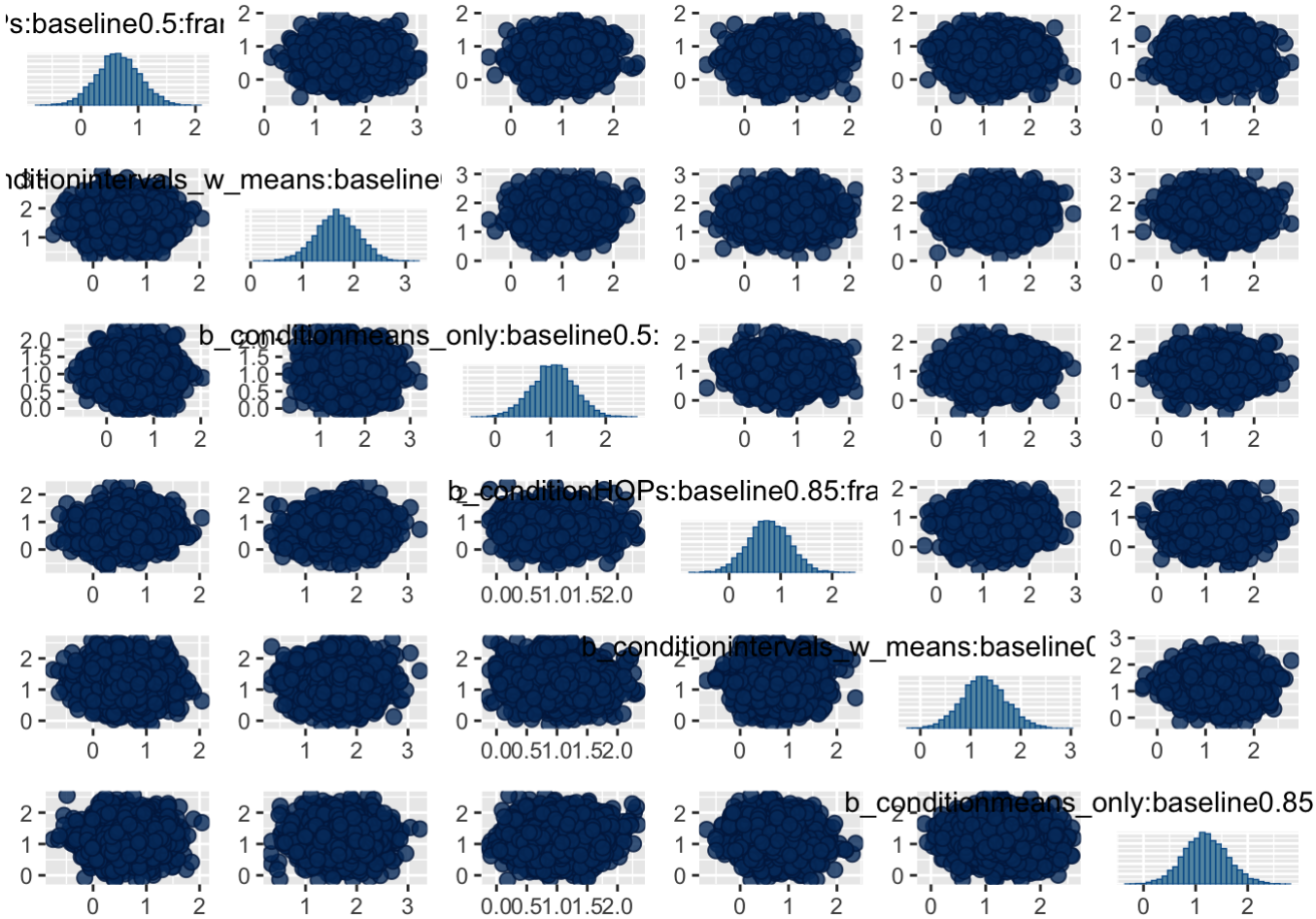


- Pairs plot

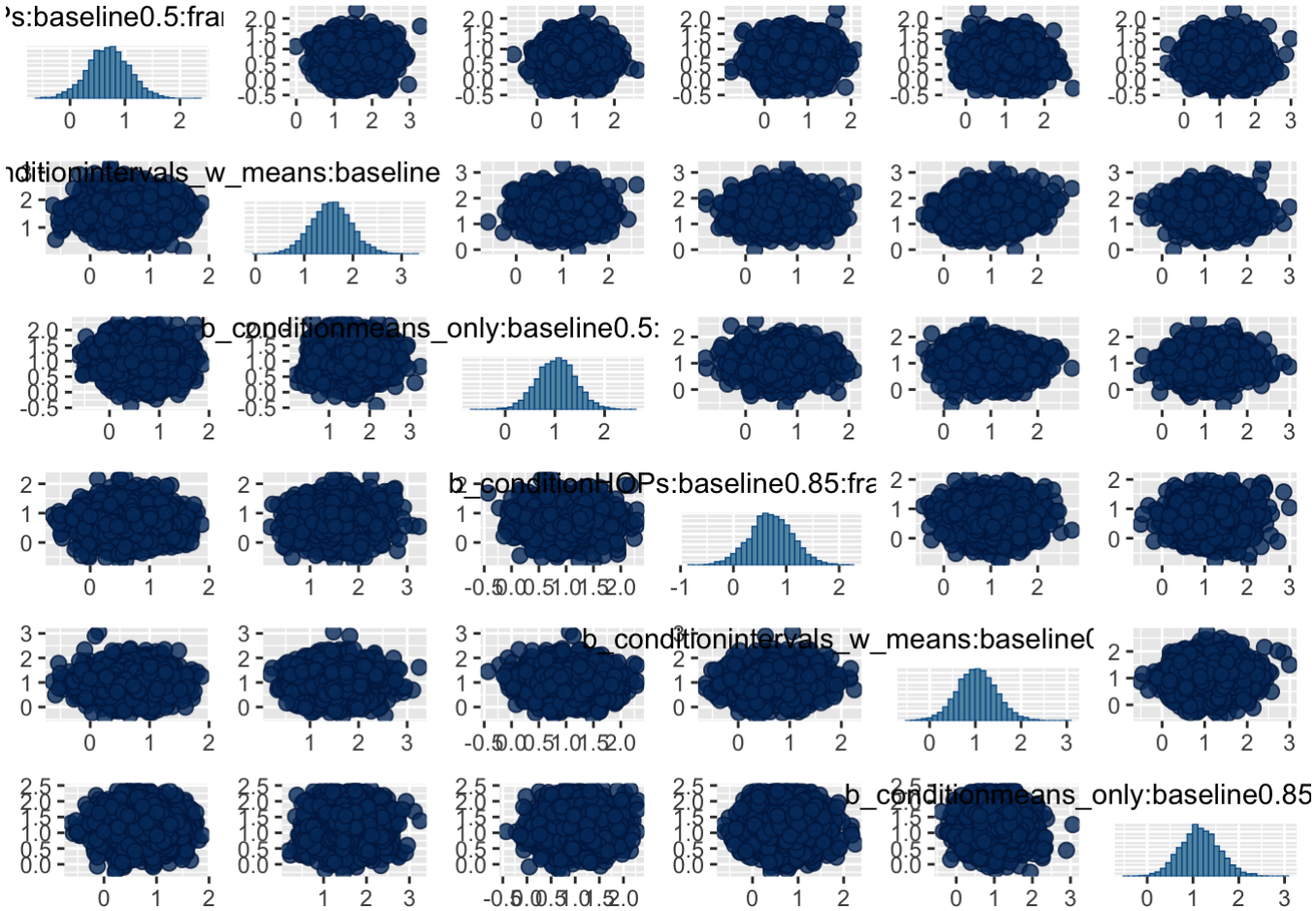
```
# pairs plot (too many things to view at once, so we've grouped them)
# hyperparameters
pairs(m.vis.base.frame.wrkr.probit, pars = c("sd_worker_id__Intercept",
                                             "sd_worker_id__evidence",
                                             "cor_worker_id__Intercept__evidence"))
```



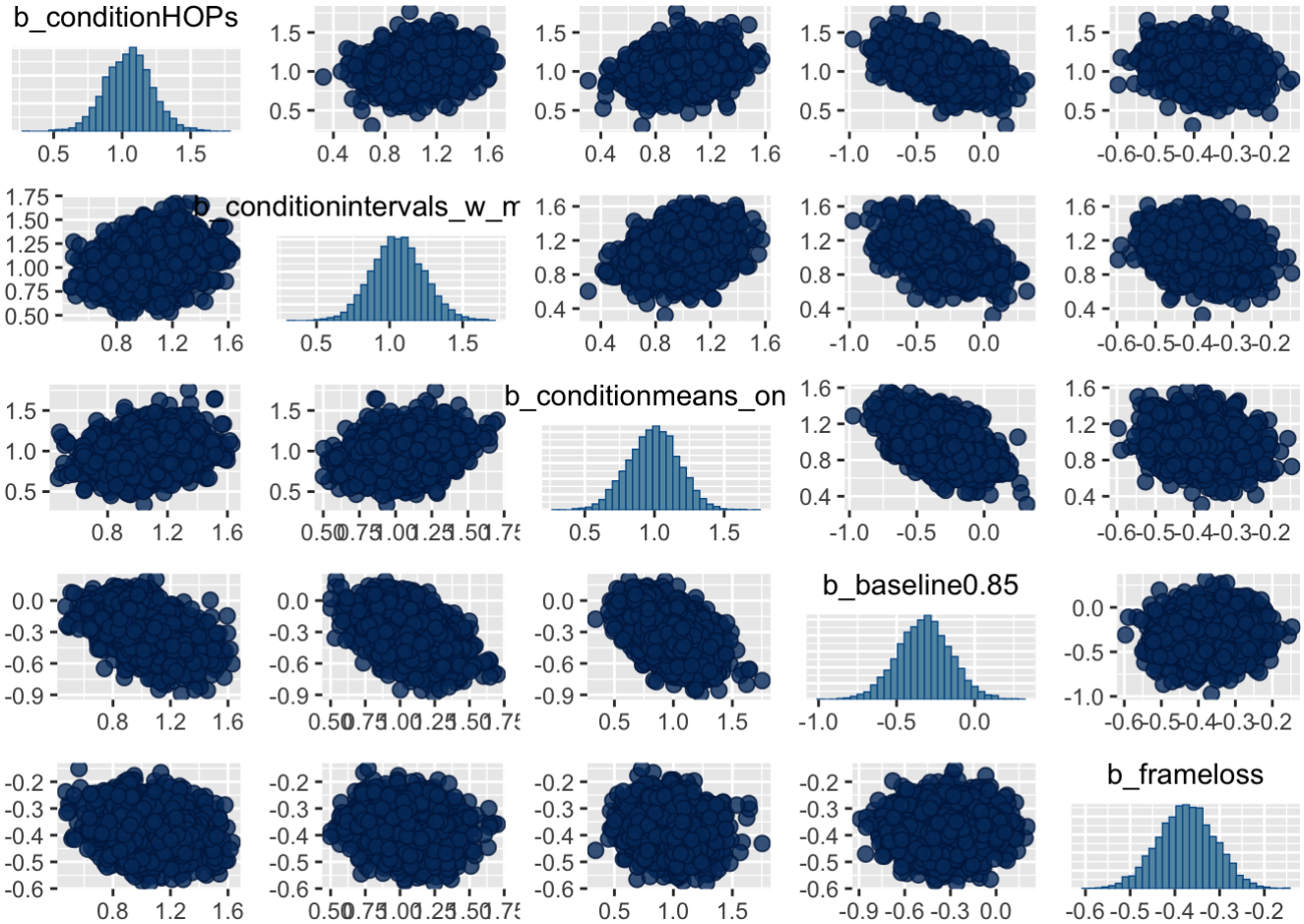

```
# pairs plot (too many things to view at once, so we've grouped them)
# slope effects: gain frame
pairs(m.vis.base.frame.wrkr.probit, pars = c("b_conditionHOPs:baseline0.5:framegain:e
vidence",
                                             "b_conditionintervals_w_means:baseline0.
5:framegain:evidence",
                                             "b_conditionmeans_only:baseline0.5:frame
gain:evidence",
                                             "b_conditionHOPs:baseline0.85:framegain:
evidence",
                                             "b_conditionintervals_w_means:baseline0.
85:framegain:evidence",
                                             "b_conditionmeans_only:baseline0.85:fram
egain:evidence"))
```




```
# pairs plot (too many things to view at once, so we've grouped them)
# slope effects: loss frame
pairs(m.vis.base.frame.wrkr.probit, pars = c("b_conditionHOPs:baseline0.5:frameloss:e
vidence",
                                             "b_conditionintervals_w_means:baseline0.
5:frameloss:evidence",
                                             "b_conditionmeans_only:baseline0.5:frame
loss:evidence",
                                             "b_conditionHOPs:baseline0.85:frameloss:
evidence",
                                             "b_conditionintervals_w_means:baseline0.
85:frameloss:evidence",
                                             "b_conditionmeans_only:baseline0.85:fram
eloss:evidence"))
```




```
# pairs plot (too many things to view at once, so we've grouped them)
# intercept effects
pairs(m.vis.base.frame.wrkr.probit, exact_match = TRUE, pars = c("b_conditionHOPs",
                                                                 "b_conditioninterval
s_w_means",
                                                                 "b_conditionmeans_on
ly",
                                                                 "b_baseline0.85",
                                                                 "b_frameloss"))
```



- Summary

```
# model summary
print(m.vis.base.frame.wrkr.probit)
```

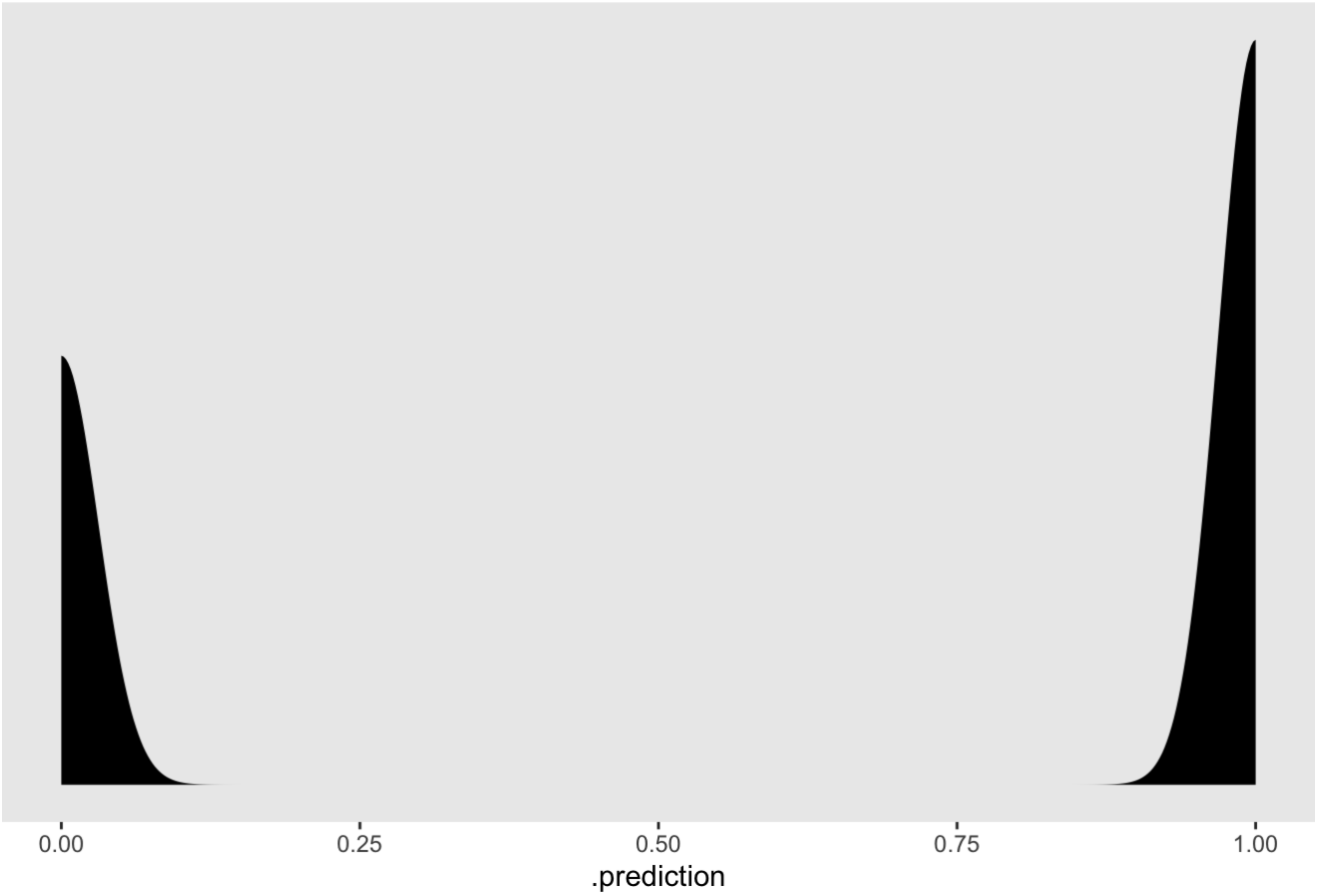
```
##  Family: bernoulli
##   Links: mu = probit
## Formula: intervene ~ (1 + evidence | worker_id) + 0 + condition + baseline + frame
+ evidence:condition:baseline:frame
##    Data: model_df (Number of observations: 2616)
## Samples: 2 chains, each with iter = 3000; warmup = 500; thin = 1;
##          total post-warmup samples = 5000
##
## Group-Level Effects:
## ~worker_id (Number of levels: 109)
##                         Estimate Est.Error l-95% CI u-95% CI Eff.Sample
## sd(Intercept)               0.85      0.08     0.69     1.02       1795
## sd(evidence)                1.65      0.24     1.22     2.18       1250
## cor(Intercept,evidence)     0.20      0.13    -0.07     0.45       1215
##                         Rhat
## sd(Intercept)           1.00
## sd(evidence)            1.00
## cor(Intercept,evidence) 1.00
##
## Population-Level Effects:
##                                                          Estimate
## conditionHOPs                                                1.05
## conditionintervals_w_means                                   1.06
## conditionmeans_only                                          0.99
## baseline0.85                                                -0.32
## frameloss                                                   -0.37
## conditionHOPs:baseline0.5:framegain:evidence                 0.65
## conditionintervals_w_means:baseline0.5:framegain:evidence    1.67
## conditionmeans_only:baseline0.5:framegain:evidence           1.05
## conditionHOPs:baseline0.85:framegain:evidence                0.77
## conditionintervals_w_means:baseline0.85:framegain:evidence   1.27
## conditionmeans_only:baseline0.85:framegain:evidence          1.19
## conditionHOPs:baseline0.5:frameloss:evidence                 0.70
## conditionintervals_w_means:baseline0.5:frameloss:evidence    1.56
## conditionmeans_only:baseline0.5:frameloss:evidence           1.03
## conditionHOPs:baseline0.85:frameloss:evidence                0.71
## conditionintervals_w_means:baseline0.85:frameloss:evidence   1.03
## conditionmeans_only:baseline0.85:frameloss:evidence          1.15
##                                                          Est.Error
## conditionHOPs                                                 0.17
## conditionintervals_w_means                                    0.18
## conditionmeans_only                                           0.18
## baseline0.85                                                  0.17
## frameloss                                                     0.06
## conditionHOPs:baseline0.5:framegain:evidence                  0.38
## conditionintervals_w_means:baseline0.5:framegain:evidence     0.41
## conditionmeans_only:baseline0.5:framegain:evidence            0.39
## conditionHOPs:baseline0.85:framegain:evidence                 0.40
## conditionintervals_w_means:baseline0.85:framegain:evidence    0.44
## conditionmeans_only:baseline0.85:framegain:evidence           0.42
## conditionHOPs:baseline0.5:frameloss:evidence                  0.38
## conditionintervals_w_means:baseline0.5:frameloss:evidence     0.41
## conditionmeans_only:baseline0.5:frameloss:evidence            0.39
## conditionHOPs:baseline0.85:frameloss:evidence                 0.41
## conditionintervals_w_means:baseline0.85:frameloss:evidence    0.43
## conditionmeans_only:baseline0.85:frameloss:evidence           0.43
##                                                          l-95% CI
```

```
## conditionHOPs                                                  0.72
## conditionintervals_w_means                                     0.72
## conditionmeans_only                                            0.64
## baseline0.85                                                  -0.66
## frameloss                                                     -0.50
## conditionHOPs:baseline0.5:framegain:evidence                  -0.08
## conditionintervals_w_means:baseline0.5:framegain:evidence      0.85
## conditionmeans_only:baseline0.5:framegain:evidence             0.27
## conditionHOPs:baseline0.85:framegain:evidence                 -0.02
## conditionintervals_w_means:baseline0.85:framegain:evidence     0.41
## conditionmeans_only:baseline0.85:framegain:evidence            0.37
## conditionHOPs:baseline0.5:frameloss:evidence                  -0.07
## conditionintervals_w_means:baseline0.5:frameloss:evidence      0.76
## conditionmeans_only:baseline0.5:frameloss:evidence             0.27
## conditionHOPs:baseline0.85:frameloss:evidence                 -0.09
## conditionintervals_w_means:baseline0.85:frameloss:evidence     0.18
## conditionmeans_only:baseline0.85:frameloss:evidence            0.33
##                                                             u-95% CI
## conditionHOPs                                                  1.39
## conditionintervals_w_means                                     1.42
## conditionmeans_only                                            1.34
## baseline0.85                                                   0.01
## frameloss                                                     -0.25
## conditionHOPs:baseline0.5:framegain:evidence                   1.40
## conditionintervals_w_means:baseline0.5:framegain:evidence      2.45
## conditionmeans_only:baseline0.5:framegain:evidence             1.80
## conditionHOPs:baseline0.85:framegain:evidence                  1.55
## conditionintervals_w_means:baseline0.85:framegain:evidence     2.16
## conditionmeans_only:baseline0.85:framegain:evidence            2.04
## conditionHOPs:baseline0.5:frameloss:evidence                   1.45
## conditionintervals_w_means:baseline0.5:frameloss:evidence      2.39
## conditionmeans_only:baseline0.5:frameloss:evidence             1.79
## conditionHOPs:baseline0.85:frameloss:evidence                  1.51
## conditionintervals_w_means:baseline0.85:frameloss:evidence     1.89
## conditionmeans_only:baseline0.85:frameloss:evidence            2.01
##                                                             Eff.Sample Rhat
## conditionHOPs                                                     2165 1.00
## conditionintervals_w_means                                        1908 1.00
## conditionmeans_only                                               1701 1.00
## baseline0.85                                                      1713 1.00
## frameloss                                                         9276 1.00
## conditionHOPs:baseline0.5:framegain:evidence                      1904 1.00
## conditionintervals_w_means:baseline0.5:framegain:evidence         2649 1.00
## conditionmeans_only:baseline0.5:framegain:evidence                2030 1.00
## conditionHOPs:baseline0.85:framegain:evidence                     2568 1.00
## conditionintervals_w_means:baseline0.85:framegain:evidence        3387 1.00
## conditionmeans_only:baseline0.85:framegain:evidence               3140 1.00
## conditionHOPs:baseline0.5:frameloss:evidence                      1774 1.00
## conditionintervals_w_means:baseline0.5:frameloss:evidence         2538 1.00
## conditionmeans_only:baseline0.5:frameloss:evidence                2068 1.00
## conditionHOPs:baseline0.85:frameloss:evidence                     2555 1.00
## conditionintervals_w_means:baseline0.85:frameloss:evidence        3423 1.00
## conditionmeans_only:baseline0.85:frameloss:evidence               3283 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's check out a posterior predictive distribution for intervention decisions.

```
# posterior predictive check
model_df %>%
  group_by(worker_id, condition, evidence) %>%
  add_predicted_draws(m.vis.base.frame.wrkr.probit, seed = 1234, n = 200) %>%
  ggplot(aes(x = .prediction)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Posterior predictive distribution for intervention") +
  theme(panel.grid = element_blank())
```

Posterior predictive distribution for intervention

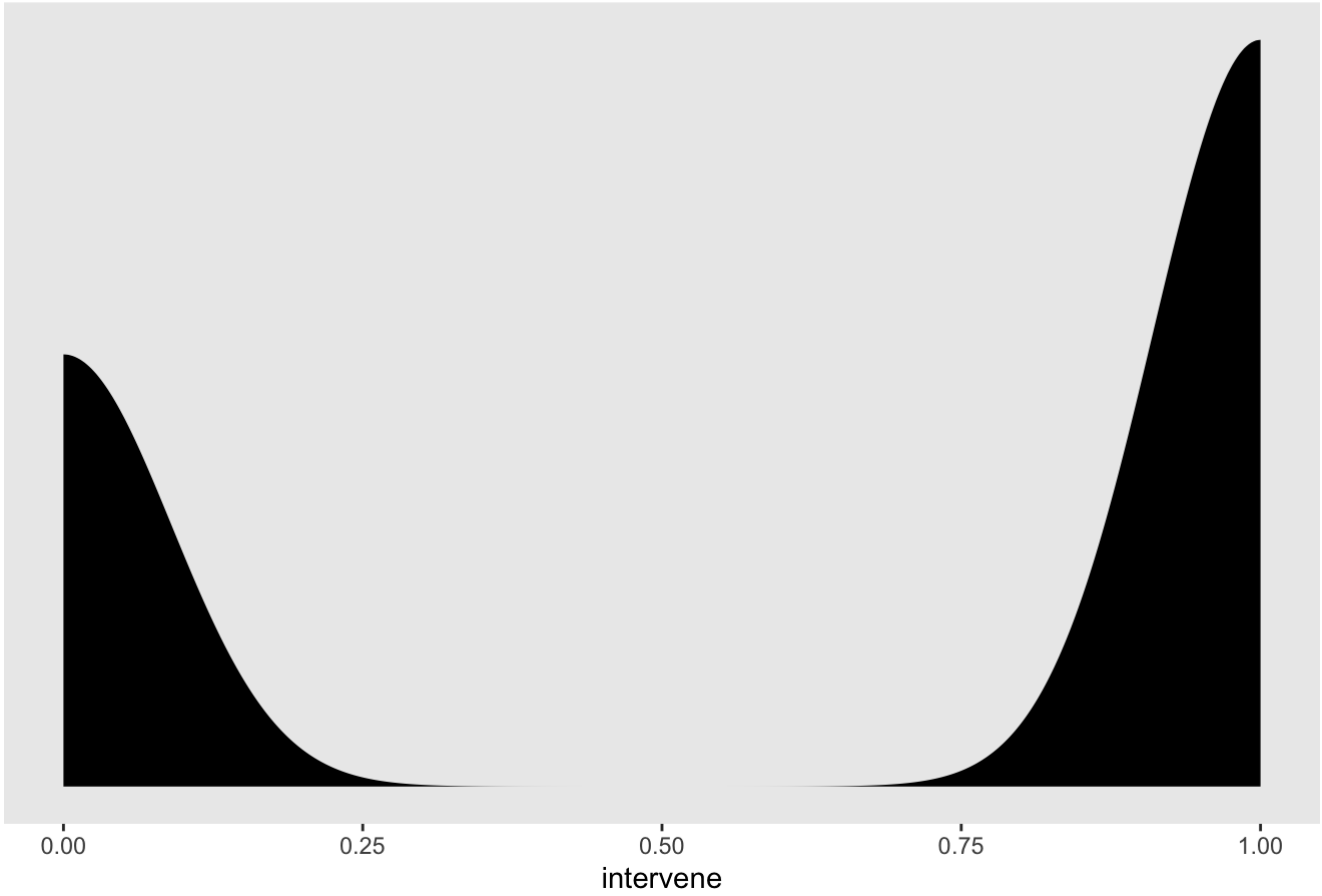


The posterior predictive distribution is about what we'd expect. The bias toward intervening is consistent with a positive intercept parameter.

How do the posterior predictions compare to the observed data?

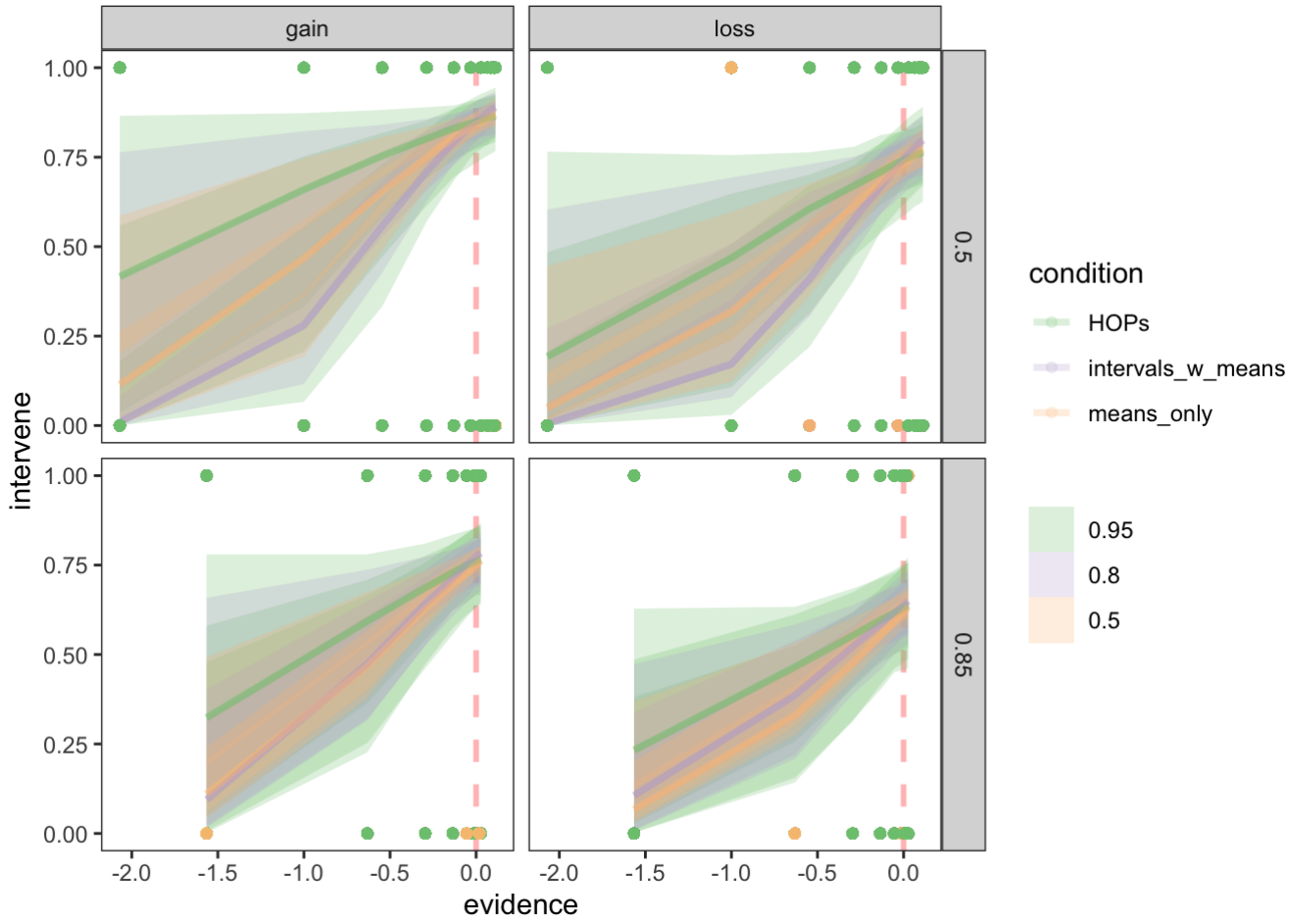
```
# data density
model_df %>%
  ggplot(aes(x = intervene)) +
  geom_density(fill = "black", size = 0) +
  scale_y_continuous(NULL, breaks = NULL) +
  labs(subtitle = "Data distribution for intervention") +
  theme(panel.grid = element_blank())
```

Data distribution for intervention



Let's take a look at the estimated psychometric functions.

```
model_df %>%
  group_by(evidence, condition, baseline, frame, worker_id) %>%
  add_fitted_draws(m.vis.base.frame.wrkr.probit, value = "pf", re_formula = NA, n = 2
00) %>%
  ggplot(aes(x = evidence, y = intervene, color = condition)) +
  geom_vline(xintercept = 0, size = 1, alpha = .3, color = "red", linetype = "dashed"
) + # utility optimal decision rule
  stat_lineribbon(aes(y = pf), .width = c(.95, .80, .50), alpha = .25) +
  geom_point(alpha = .15) +
  scale_fill_brewer(type = "qual", palette = 1) +
  scale_color_brewer(type = "qual", palette = 1) +
  coord_cartesian(xlim = quantile(model_df$evidence, c(0, 1)),
                  ylim = quantile(model_df$intervene, c(0, 1))) +
  theme_bw() +
  theme(panel.grid = element_blank()) +
  facet_grid(baseline ~ frame)
```



Again we see a slight difference in psychometric functions for visualization conditions across problem frames and levels of baseline, with more convergence of performance in the loss frame and baseline == 0.85.

Model Comparison

Let's check check which of these hierarchical models fits best insofar as the parameters contribute more to predictive validity than they contribute to overfitting. We'll determine this by comparing the models according to the widely applicable information criterion (WAIC). Lower values of WAIC indicate a better fitting model.

```
waic(m.vis.wrkr.probit, m.vis.base.wrkr.probit, m.vis.frame.wrkr.probit, m.vis.base.frame.wrkr.probit)
```

```
##                                                      WAIC    SE
## m.vis.wrkr.probit                                2551.30 51.42
## m.vis.base.wrkr.probit                           2549.94 51.35
## m.vis.frame.wrkr.probit                          2519.47 51.52
## m.vis.base.frame.wrkr.probit                     2527.70 51.54
## m.vis.wrkr.probit - m.vis.base.wrkr.probit          1.36  1.87
## m.vis.wrkr.probit - m.vis.frame.wrkr.probit        31.83 12.32
## m.vis.wrkr.probit - m.vis.base.frame.wrkr.probit   23.61 12.48
## m.vis.base.wrkr.probit - m.vis.frame.wrkr.probit   30.47 12.45
## m.vis.base.wrkr.probit - m.vis.base.frame.wrkr.probit 22.25 12.40
## m.vis.frame.wrkr.probit - m.vis.base.frame.wrkr.probit -8.23  2.12
```

A couple observations are important here:

1. The model with the lowest WAIC value (i.e., the best fitting model) is the one with predictors for visualization condition and problem frameing but not baseline condition.

2. The model with predictors for both baseline and frame has a higher WAIC value than the model with predictors for frame but not baseline. This means that adding predictors for the effect of baseline to the model with predictors for framing contributes less to predictive validity than it does to overfitting.

The conclusion I draw from this is that we don't see to learn much by manipulating the baseline condition, but problem framing is important. consistent with prior work in behavioral economics.