Supplementary Materials: Search strategies improve with practice, but not with time pressure or financial incentives

ADF Clarke

30/06/2020

Contents

These supplementary materials contain more details of the Bayesian analysis, including power analysis. Please see the source Rmd file for full code.

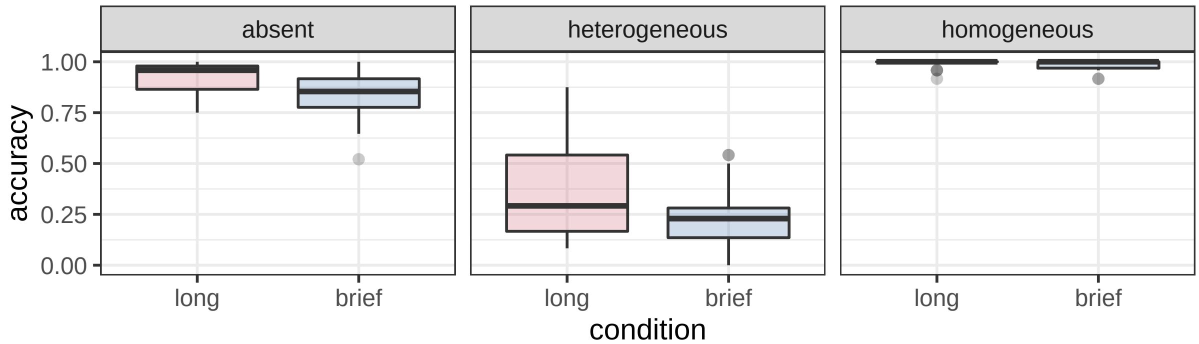
1 Experiment 1: Deadline

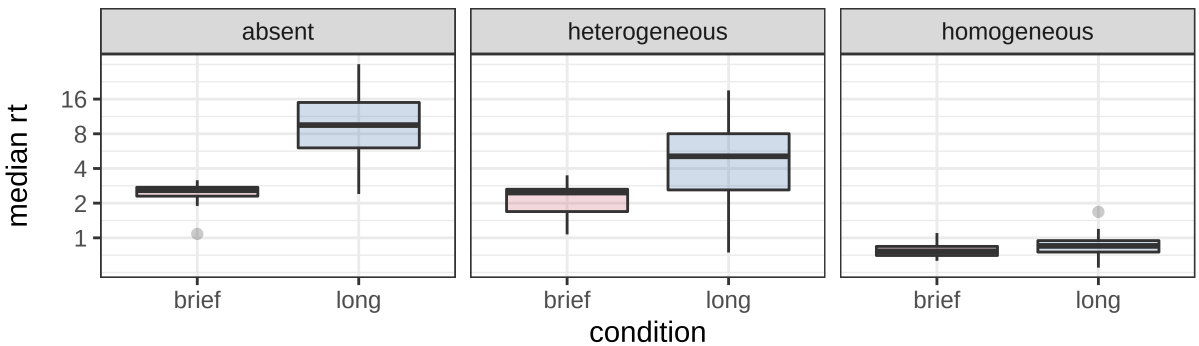
1.1 Descriptive Statistics

We will first look at descriptive statistics for accuracy and reaction time data, to check that it looks sensible and inline with our expectations.

1.1.1 Accuracy and Reaction Time

After plotting the accuracy data, incorrect trials are removed from all further analysis.



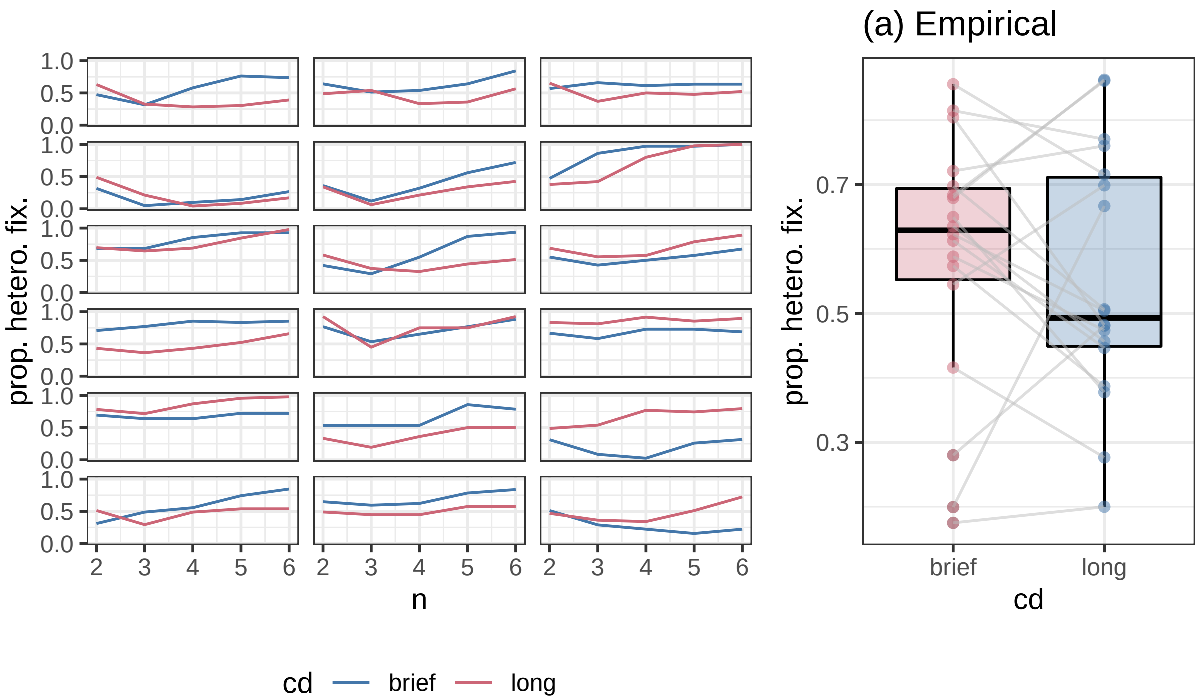


1.1.2 Saccadic Strategy

First, we need to merge (join) the fixation and accuracy data, so that we can take only correct target absent trials. We will compute the proportion of fixations to the heterogeneous side of the display for each fixation number, over all trials made by a participant.

Create a facet plot of each individual's strategy.

We can further summarise the data by creating a strategy measure, which is the proportion of all (2 - 6) fixations made by an observer over all trials.



1.2 Bayesian Model of Saccadic Strategy

Summarise data so that we have one strategy score per trial per observer.

Note, as beta distributions are only defined over (0, 1), values of 0 and 1 are impossible. To get around this, we will set any such values to 0.001 and 0.999 respectively.

1.2.1 Define function for plotting model output

I will want to reuse this plotting code, so I will put it in a function here.

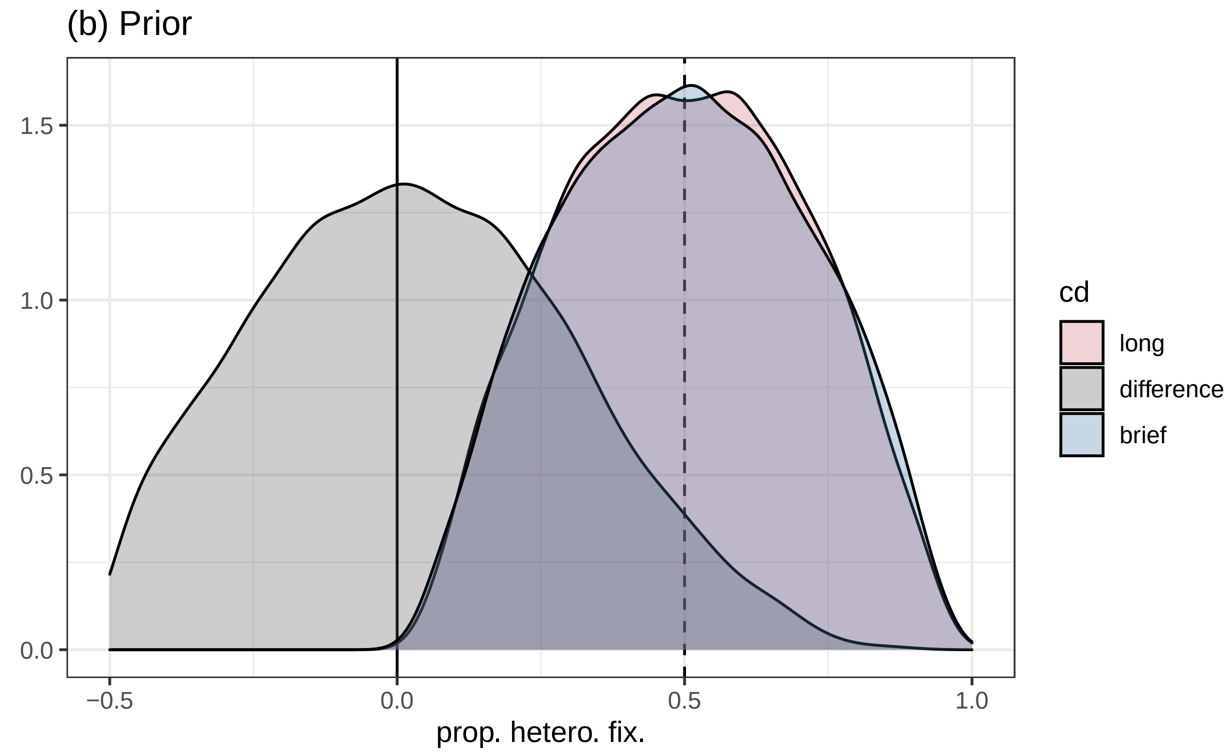
1.2.2 Define Priors

We will use $N(0,1)$, weakly informative priors, illustrated in the plot below.

```
model_priors <- c(
    prior(normal(0, 1), class = "b"))
```

```
prior_model <- brm(
    data = d_strat,
    prop_hetero ~ 0 + cd + (cd | observer),
    family = "beta",
    sample_prior = "only",
    prior = model_priors,
    iter = 5000,
    control = list(adapt_delta = 0.95))
```

```
## Warning: Removed 497 rows containing non-finite values (stat_density).
```



1.2.3 Power Analysis

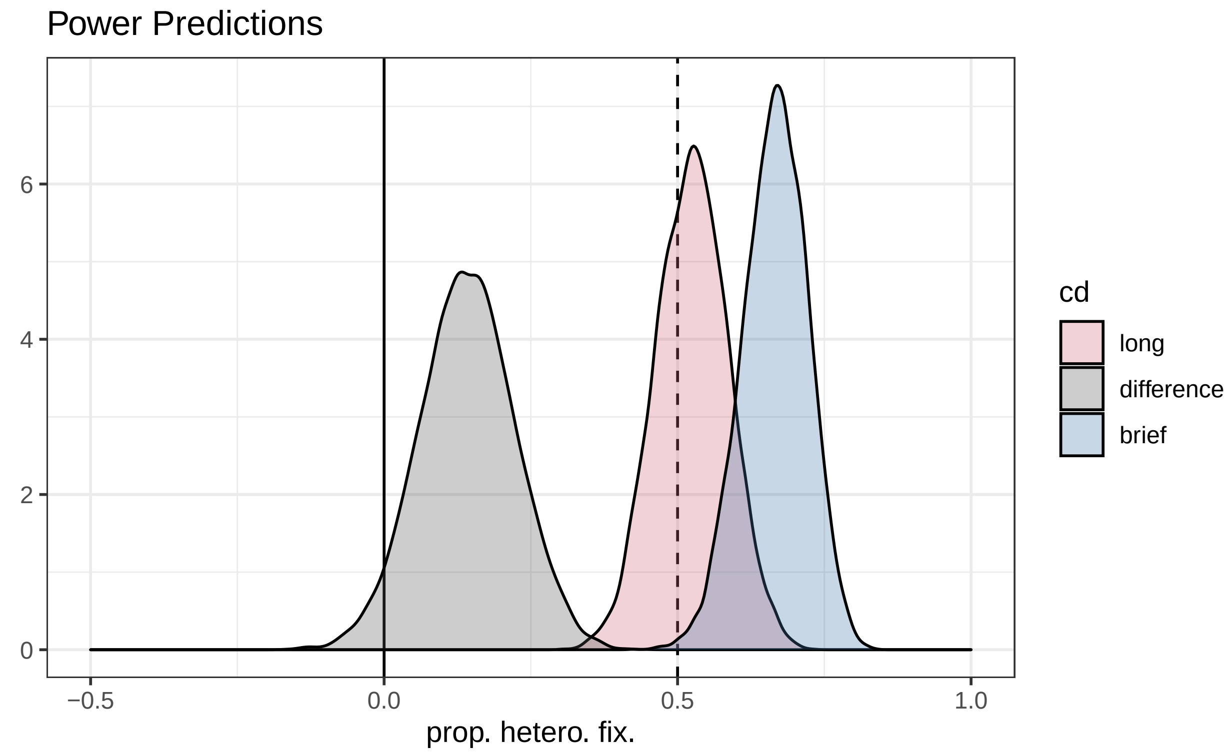
We will carry out our power analysis by simulating our experiment assuming the distributions below.

```
power_prior <- c(
  prior(normal(0.1, 0.25), class = "b", coef = "cdlong"),
    prior(normal(0.7, 0.25), class = "b", coef = "cdbrief"),
  prior(student_t(3, 0, 2), class = "sd"),
  prior(gamma(1, 10), class = "phi")
  )
```

```
## Compiling Stan program...
```

```
## Start sampling
```

We can now plot these distributions to check that they seem reasonable.



These corresponds to assuming distributions with the means presented below:

cd	mean prop. fix hetero.
brief	0.6056265
long	0.5175624

We now generate multiple (=50) simulated datasets with 15 observers and 32 correct target absent trials. We then compute $p(\delta > 0|d)$, (the probablity, given the data, of seeing postive difference between the brief and long conditions) for each. The expected distribution of this statistic is shown below. We can see that in (almost?) every iteration, we get a value about 0.95.

And plot!

1.3 Compute and Plot Posterior

Now that we are confident that we have a sensible prior, and have carried out a power analysis, it is time to fit the model to the data.

```
my_model <- brm(
  data = d_strat,
  prop_hetero ~ 0 + cd + (cd | observer),
  family = "beta",
    prior = model_priors,
    iter = 10000,
    control = list(adapt_delta = 0.95))
```

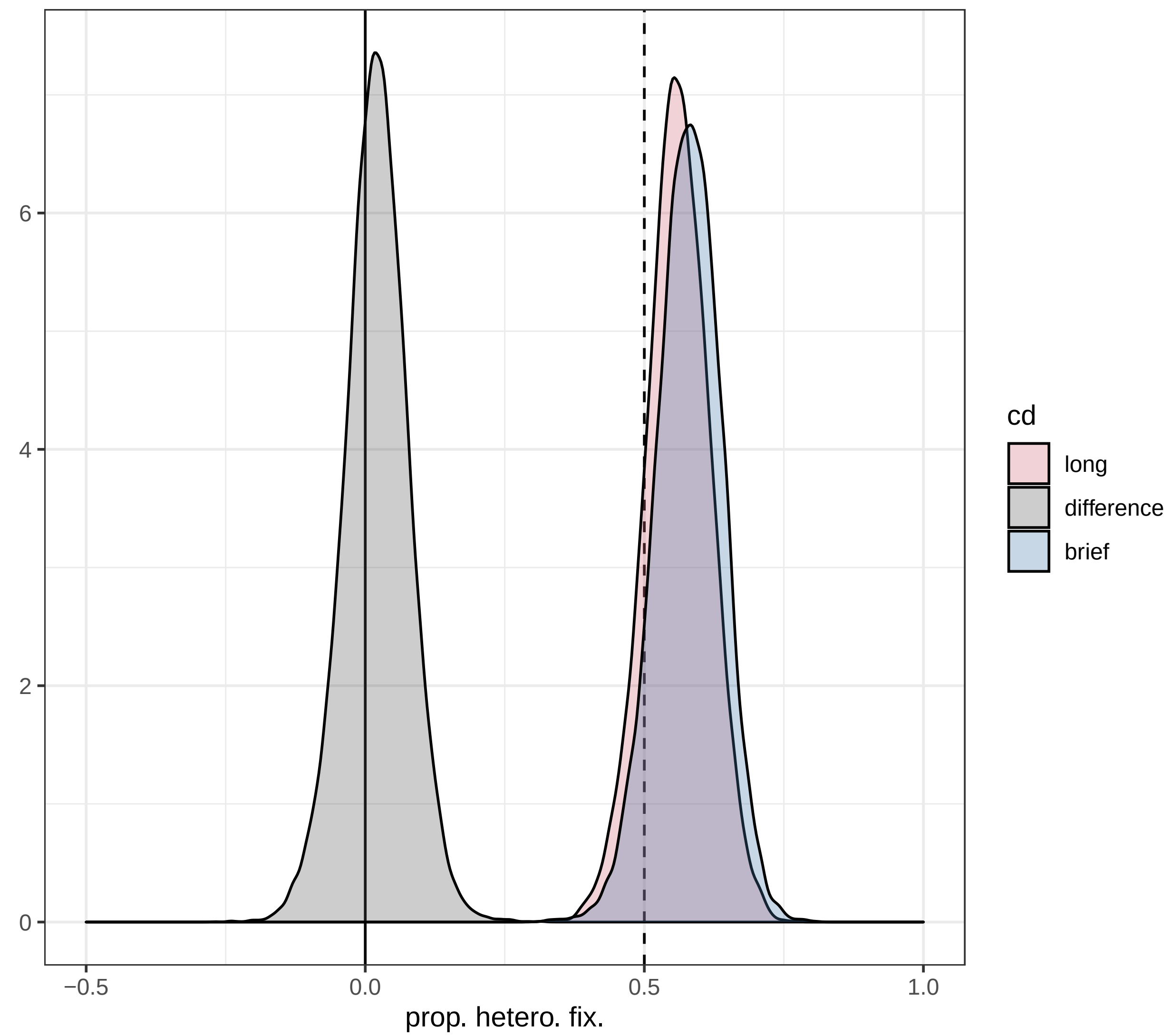
```
## Compiling Stan program...
```

```
## Start sampling
```

```
##  Family: beta
##   Links: mu = logit; phi = identity
```

```
## Formula: prop_hetero ~ 0 + cd + (cd | observer)
##    Data: d_strat (Number of observations: 1520)
## Samples: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
##          total post-warmup samples = 20000
##
## Group-Level Effects:
## ~observer (Number of levels: 18)
##                        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)              1.05      0.20     0.74     1.53 1.00     4795     8638
## sd(cdlong)                 0.99      0.20     0.68     1.44 1.00     4876     8582
## cor(Intercept,cdlong)     -0.49      0.19    -0.79    -0.06 1.00     5168     7770
##
## Population-Level Effects:
##         Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## cdbrief     0.32      0.24    -0.16     0.80 1.00     3896     6054
## cdlong      0.24      0.24    -0.23     0.71 1.00     4735     7634
##
## Family Specific Parameters:
##     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## phi     1.71      0.06     1.60     1.83 1.00    17530    14646
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

(c) Posterior




```
## Warning: Removed 497 rows containing non-finite values (stat_density).
```

long	long.lower	long.upper	brief	brief.lower	brief.upper	difference	difference.lower	difference.upp
0.5596482	0.443425	0.6697834	0.5806241	0.4658396	0.6944334	0.0202445	-0.0971617	0.131484

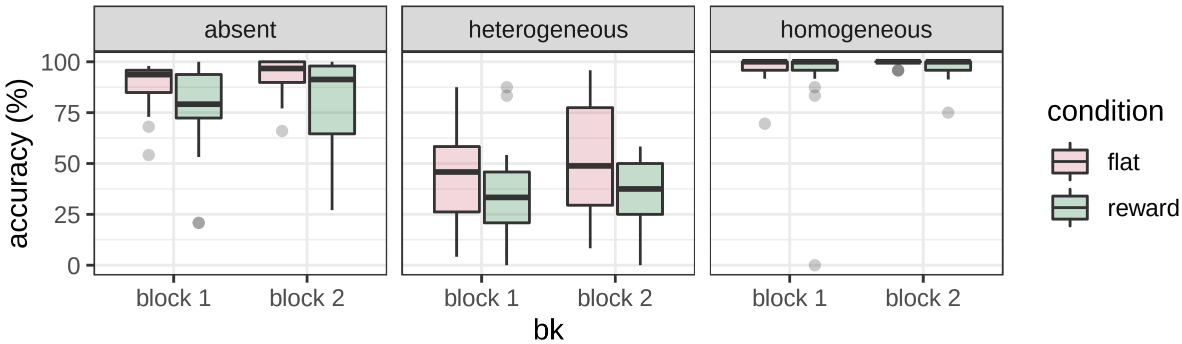
We can conclude that $p(x>0 | d) = $ 0.6463.

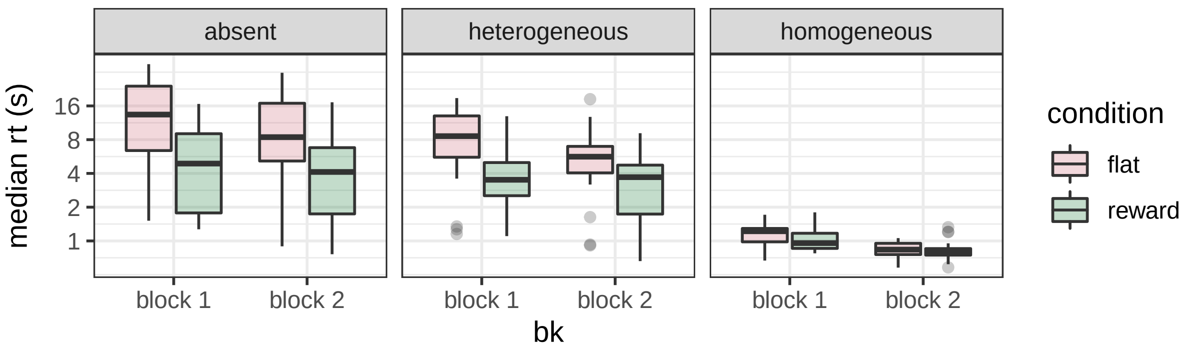
2 Experiment 2: Reward

2.1 Descriptive Statistics

2.1.1 Accuracy and Reaction Time

After plotting the accuracy data, incorrect trials are removed from all further analysis.





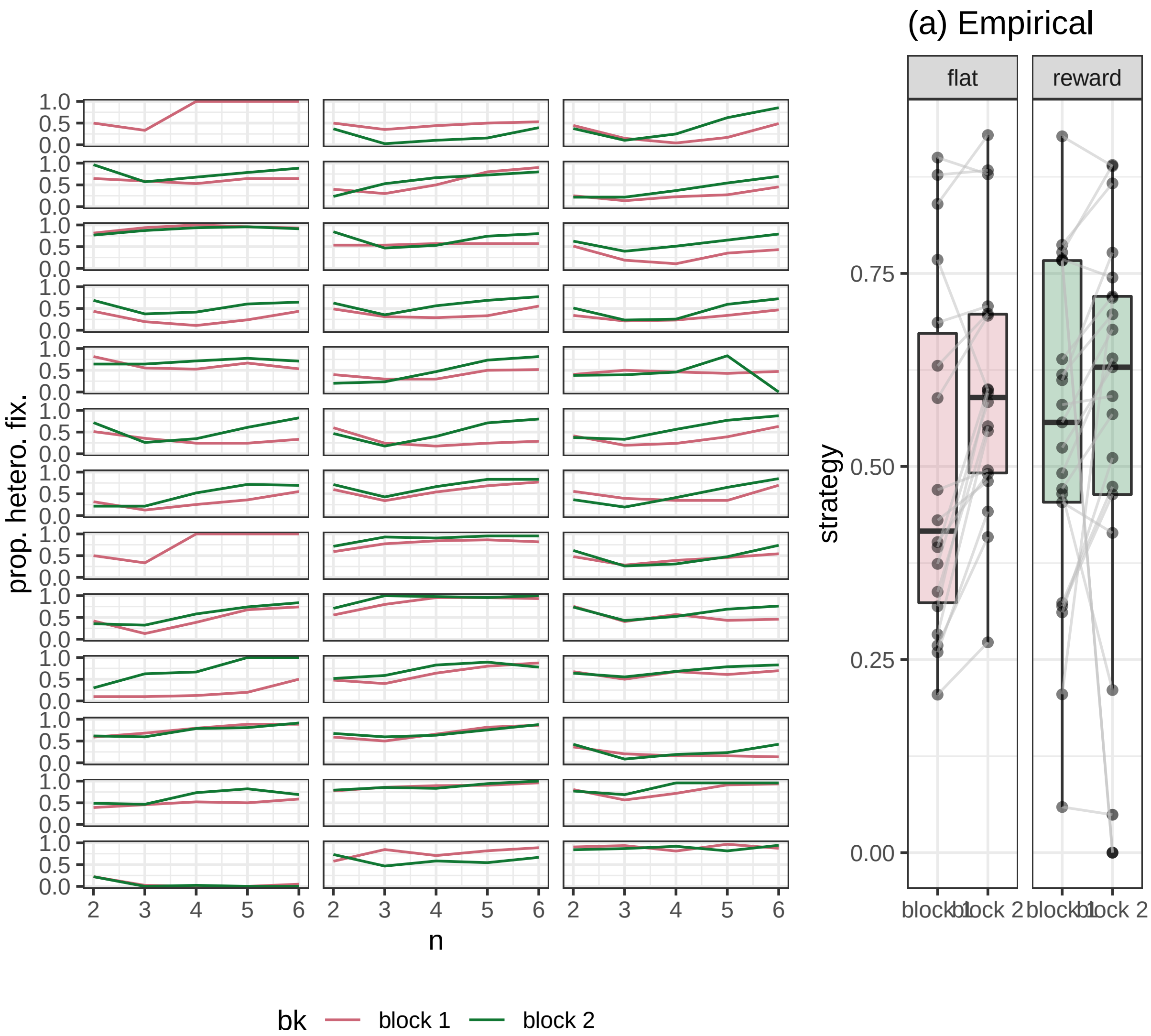
2.2 Saccadic Strategy

First, we need to merge (join) the fixation and accuracy data, so that we can take only correct target absent trials. We will compute the proportion of fixations to the heterogeneous side of the display for each fixation number, over all trials made by a participant.

Create a facet plot of each individual's strategy.

We can further summarise the data by creating a strategy measure, which is the proportion of all (2 - 6) fixations made by an observer over all trials.

```
## `summarise()` has grouped output by 'observer', 'bk'. You can override using the `.groups` argument.
```



2.3 Bayesian Model of Saccadic Strategy

Summarise data so tha we have one strategy score per trial per observer.

Note, as beta distributions are only defined over (0, 1), values of 0 and 1 are impossible. To get around this, we will set any such values to 0.001 and 0.999 respectively.

2.3.1 Define Priors

We will use the same priors, and model structure, as above.

```
model_priors <- c(
    prior(normal(0, 1), class = "b"))
```

```
prior_model <- brm(
    data = d_strat,
    prop_hetero ~ 0 + cd + (cd | observer),
    sample_prior = "only",
```

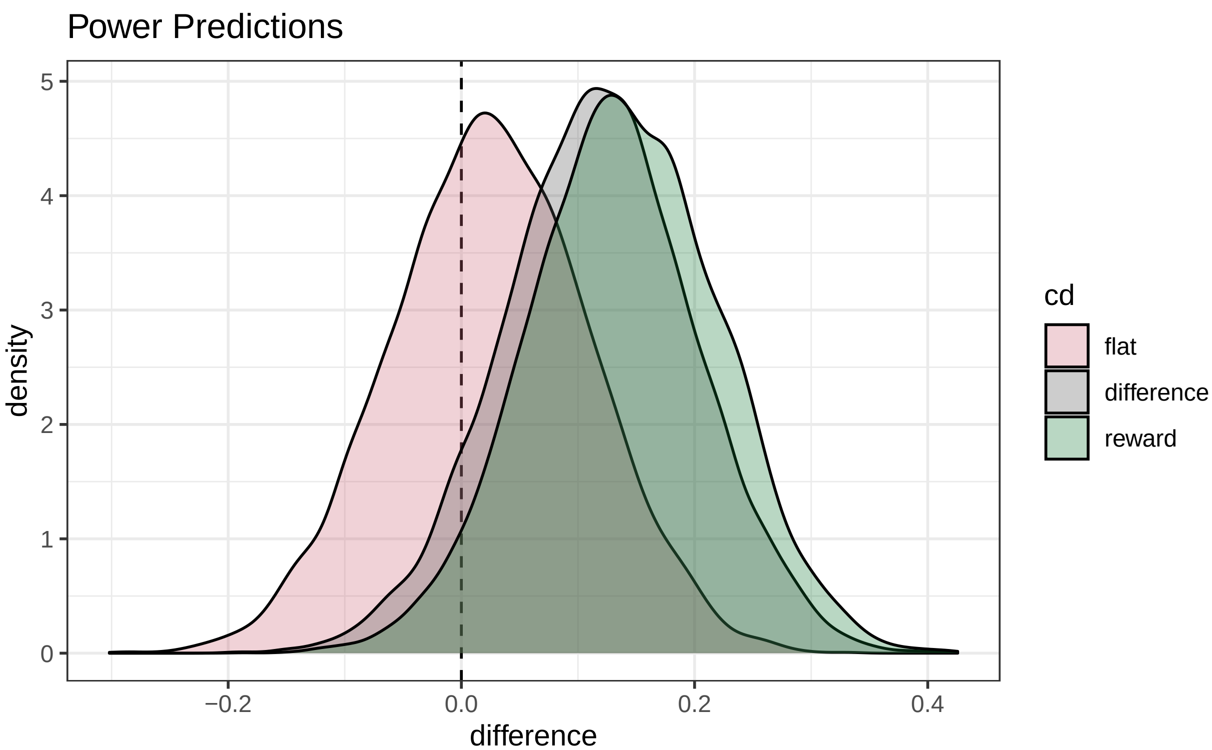
```
    family = "beta",
    prior = model_priors,
    iter = 10000,
    control = list(adapt_delta = 0.95))
```

And plot, to see if it looks reasonable.

2.3.2 Power Analysis

We will carry out our power analysis by simulating our experiment assuming the distributions below.

We can now plot these distributions to check that they seem reasonable.

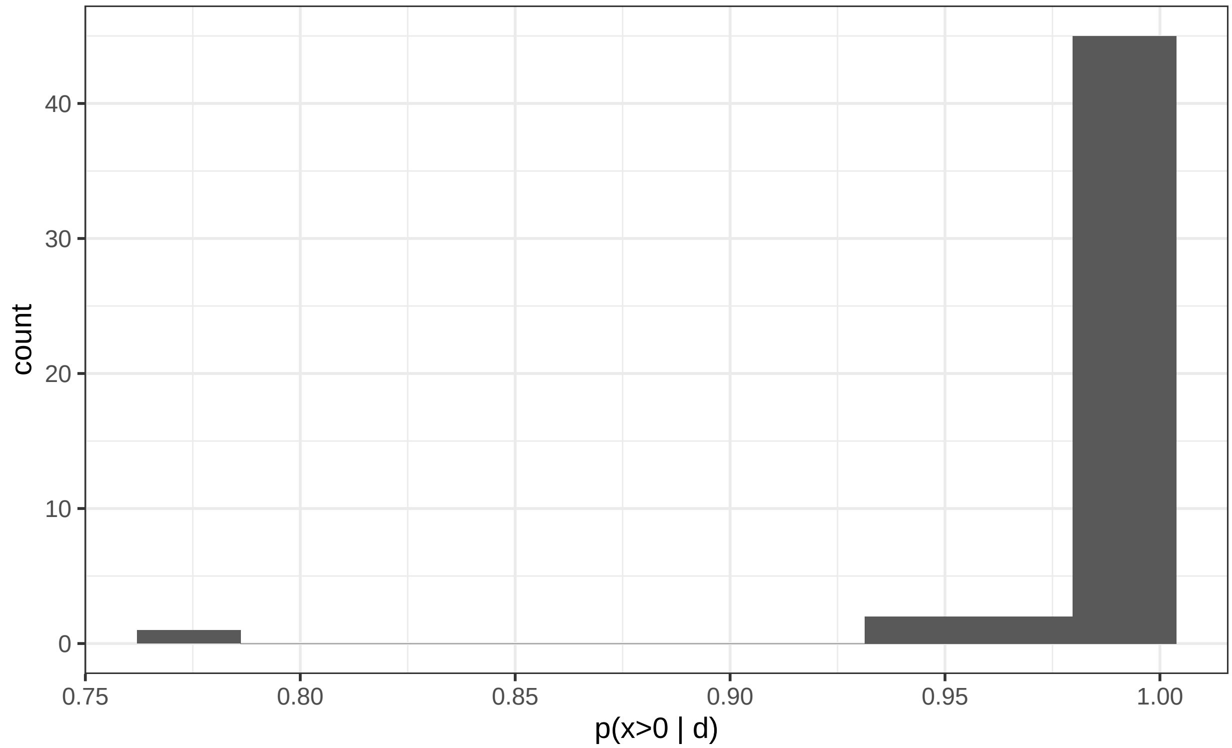


Next, we write a function to generate a simulated dataset.

Now we also need a function that will the key statistic that we are interested in: the probability, given the data, that observers were more strategic in the brief condition than the long.

Finally, we run this a number of times (50) to see the distribution of $p(\delta > 0|d)$ assuming 15 observers and 32 correct target absent trials.

And plot!



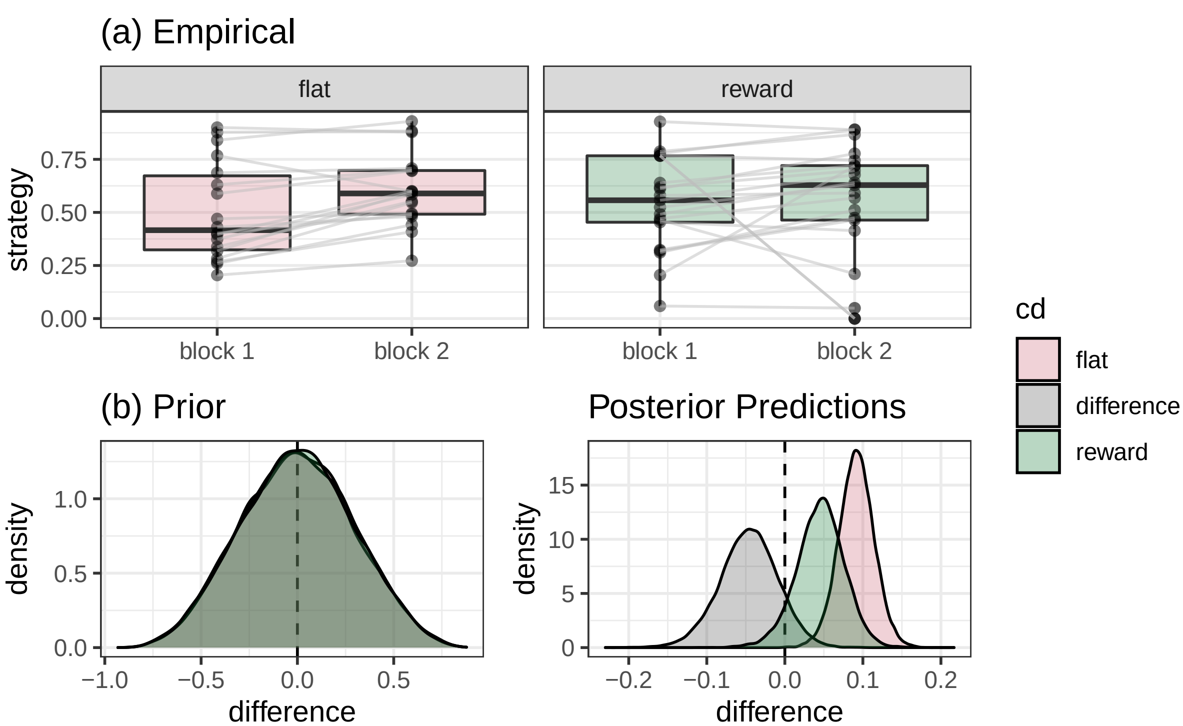

```
## # A tibble: 1 x 1
##   over_90
##     <int>
## 1      49
```

```
## # A tibble: 3 x 2
##   cd      mean_ph
## * <chr>     <dbl>
## 1 flat      0.518
## 2 initial   0.496
## 3 reward    0.590
```

2.4 Now refit model using data

```
##  Family: beta
##   Links: mu = logit; phi = identity
## Formula: prop_hetero ~ 0 + cd + (cd | observer)
##    Data: d_strat (Number of observations: 3025)
## Samples: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
##          total post-warmup samples = 20000
##
## Group-Level Effects:
## ~observer (Number of levels: 39)
##                         Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)               0.87      0.11     0.69     1.12 1.00     5175
## sd(cdreward)                0.48      0.14     0.26     0.80 1.00     7756
## sd(cdflat)                  0.35      0.09     0.18     0.55 1.00    10343
## cor(Intercept,cdreward)     0.01      0.25    -0.45     0.49 1.00    12577
## cor(Intercept,cdflat)      -0.57      0.19    -0.87    -0.12 1.00    16511
## cor(cdreward,cdflat)       -0.01      0.47    -0.84     0.83 1.00     3411
```

```
##                           Tail_ESS
## sd(Intercept)                 8915
## sd(cdreward)                 11091
## sd(cdflat)                   11936
## cor(Intercept,cdreward)      14330
## cor(Intercept,cdflat)        14769
## cor(cdreward,cdflat)          9567
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## cdinitial     0.01      0.14    -0.26     0.29 1.00     2907     5646
## cdreward      0.20      0.18    -0.16     0.54 1.00     4262     9061
## cdflat        0.38      0.13     0.12     0.65 1.00     4081     7703
##
## Family Specific Parameters:
##     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## phi     1.86      0.04     1.78     1.95 1.00    31177    14260
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```



What is the the probability of a difference > 0, given the data?

```
## # A tibble: 1 x 1
##   prob_diff_greater_zero
##                    <dbl>
## 1                  0.104
```

```
## # A tibble: 1 x 12
##     flat flat.lower flat.upper reward reward.lower reward.upper difference
```

```
##    <dbl>      <dbl>      <dbl>  <dbl>        <dbl>        <dbl>      <dbl>
## 1 0.0918     0.0447      0.135 0.0463      -0.0177        0.104    -0.0454
## # ... with 5 more variables: difference.lower <dbl>, difference.upper <dbl>,
## #   .width <dbl>, .point <chr>, .interval <chr>
```

2.5 Original Pre-Registered Analysis

We original pre-registered an analysis plan for this experiment using frequentist statistics. The results of this planned analysis are presented here.

2.5.1 Accuracy

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: accuracy ~ bk * cd + (1 | observer)
##    Data: d_lmer_acc
##
## REML criterion at convergence: -137.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.7596 -0.3331  0.0032  0.4835  2.3775
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  observer (Intercept) 0.006457 0.08036
##  Residual             0.003678 0.06065
## Number of obs: 78, groups:  observer, 39
##
## Fixed effects:
##                     Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)         0.796026   0.023729 52.634479  33.547  < 2e-16 ***
## bkblock 2           0.043949   0.020215 37.000000   2.174  0.03617 *
## cdreward           -0.091066   0.032337 52.634479  -2.816  0.00683 **
## bkblock 2:cdreward -0.009082   0.027549 37.000000  -0.330  0.74350
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) bkblc2 cdrwrd
## bkblock 2   -0.426
## cdreward    -0.734  0.313
## bkblck2:cdr  0.313 -0.734 -0.426

## Type III Analysis of Variance Table with Satterthwaite's method
##         Sum Sq  Mean Sq NumDF DenDF F value   Pr(>F)
## bk    0.030104 0.030104     1    37  8.1851 0.006907 **
## cd    0.039275 0.039275     1    37 10.6788 0.002344 **
## bk:cd 0.000400 0.000400     1    37  0.1087 0.743499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.5.2 Median Reaction Time

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
```

```
## lmerModLmerTest]
## Formula: median_rt ~ bk * cd + (1 | observer)
##    Data: d_lmer_rt
##
## REML criterion at convergence: 434.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.2991 -0.3078 -0.0809  0.3153  3.2528
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  observer (Intercept) 26.257   5.124
##  Residual              5.393   2.322
## Number of obs: 78, groups:  observer, 39
##
## Fixed effects:
##                     Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)          11.4244     1.3260 43.8324   8.616 5.54e-11 ***
## bkblock 2            -3.4926     0.7741 37.0000  -4.512 6.30e-05 ***
## cdreward             -7.1036     1.8071 43.8324  -3.931 0.000297 ***
## bkblock 2:cdreward    2.9688     1.0549 37.0000   2.814 0.007786 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) bkblc2 cdrwrd
## bkblock 2   -0.292
## cdreward    -0.734  0.214
## bkblck2:cdr  0.214 -0.734 -0.292
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##       Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
## bk    78.174  78.174     1    37 14.4957 0.0005119 ***
## cd    57.004  57.004     1    37 10.5701 0.0024528 **
## bk:cd 42.713  42.713     1    37  7.9202 0.0077858 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: median_rt ~ bk * cd + (1 | observer)
##    Data: d_lmer_rt
##
## REML criterion at convergence: 434.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.2991 -0.3078 -0.0809  0.3153  3.2528
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  observer (Intercept) 26.257   5.124
##  Residual              5.393   2.322
## Number of obs: 78, groups:  observer, 39
```

```
##
## Fixed effects:
##                     Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)          11.4244     1.3260 43.8324   8.616 5.54e-11 ***
## bkblock 2            -3.4926     0.7741 37.0000  -4.512 6.30e-05 ***
## cdreward             -7.1036     1.8071 43.8324  -3.931 0.000297 ***
## bkblock 2:cdreward    2.9688     1.0549 37.0000   2.814 0.007786 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) bkblc2 cdrwrd
## bkblock 2   -0.292
## cdreward    -0.734  0.214
## bkblck2:cdr  0.214 -0.734 -0.292
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##       Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
## bk    78.174  78.174     1    37 14.4957 0.0005119 ***
## cd    57.004  57.004     1    37 10.5701 0.0024528 **
## bk:cd 42.713  42.713     1    37  7.9202 0.0077858 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.5.3 Search Efficiency

```
## fixed-effect model matrix is rank deficient so dropping 3 columns / coefficients
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: median_search_ef ~ bk * cd + (1 | observer)
##    Data: d_lmer_se
##
## REML criterion at convergence: 2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.0444 -0.4745  0.0963  0.4609  1.4285
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  observer (Intercept) 0.05182  0.2276
##  Residual             0.02207  0.1486
## Number of obs: 78, groups:  observer, 39
##
## Fixed effects:
##             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)  0.50256    0.04353 50.82578  11.546 8.07e-16 ***
## bkblock 2    0.13314    0.04749 43.21037   2.804  0.00754 **
## cdreward    -0.10718    0.06224 49.94386  -1.722  0.09125 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) bkblc2
```

```
## bkblock 2 -0.274
## cdreward   0.000 -0.706
## fit warnings:
## fixed-effect model matrix is rank deficient so dropping 3 columns / coefficients
```

```
## Missing cells for: bkblock 2:cdinitial, bkblock 1:cdreward, bkblock 1:cdflat.
## Interpret type III hypotheses with care.
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##         Sum Sq  Mean Sq NumDF  DenDF F value   Pr(>F)
## bk    0.173494 0.173494     1 43.210  7.8604 0.007539 **
## cd    0.065451 0.065451     1 49.944  2.9654 0.091255 .
## bk:cd
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.5.4 Discussion

The results are consistent with the analysis presented in the paper

3 Time-course Analysis

I will now look to see what happens to search strategy over time: both on the scale of an individual trial, within and across blocks, and between experimental condition! First, I will fit one model to the data from both experiments.

3.1 Prior Predictions

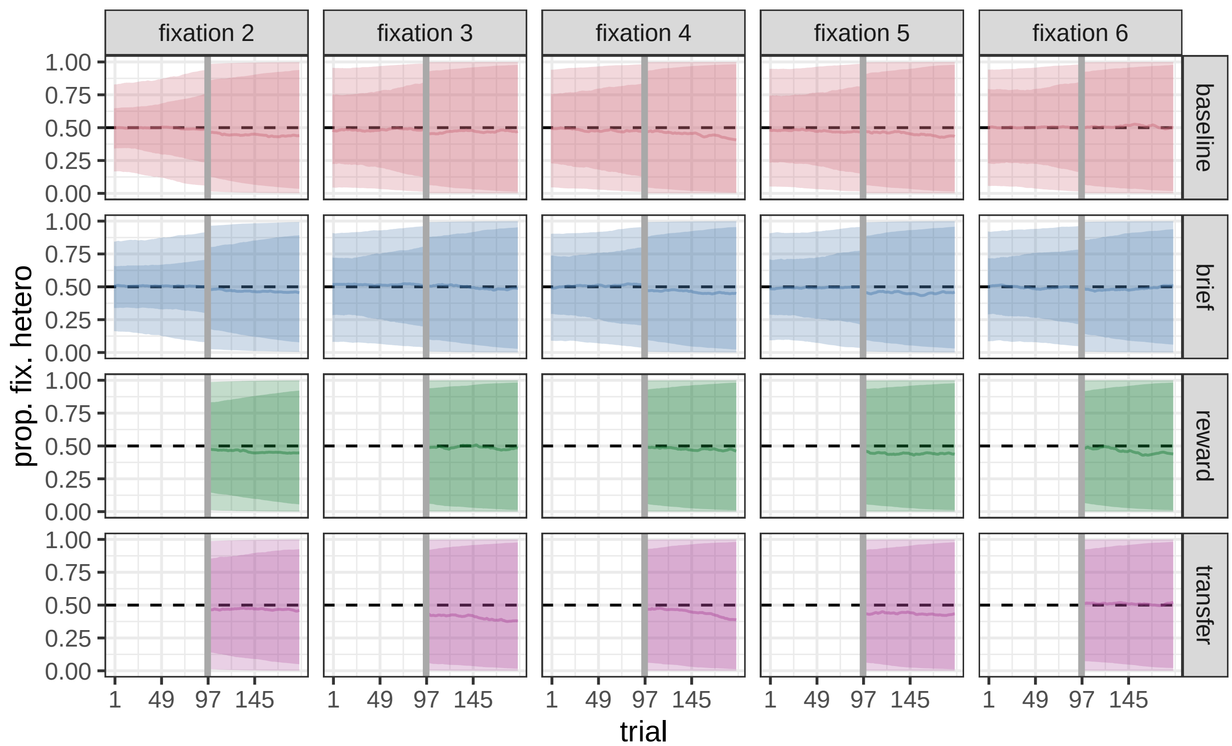
```
model_priors <- c(
    prior(normal(0, 1.0), class = "b"))

m_prior <- brm(
    data = d_strat,
    hetero_fix ~ (0 + cd) * (0 + n) * bk * ts  +
      (ts + n | observer) ,
    family = "bernoulli",
    sample_prior = "only",
    prior = model_priors,
    chains = 1)
```

```
##
## SAMPLING FOR MODEL 'a6e3cea5a4773a4134b5d4e909495233' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.444 seconds (Warm-up)
## Chain 1:                0.427 seconds (Sampling)
## Chain 1:                0.871 seconds (Total)
## Chain 1:
```

```
saveRDS(m_prior, "models/my_prior.model")
m_prior <- readRDS("models/my_prior.model")
```

```
## Warning: Removed 10220 row(s) containing missing values (geom_path).
```



3.2 Posterior Predictions

```
m_posterior <- brm(
    data = d_strat,
    hetero_fix ~ (0 + cd) * (0 + n) * bk * ts  +
     (ts + n | observer) ,
    family = "bernoulli",
    prior = model_priors,
    chains = 4)

m_posterior <- add_criterion(m_posterior, c("loo", "waic"))

saveRDS(m_posterior, "models/my_posterior.model")
```

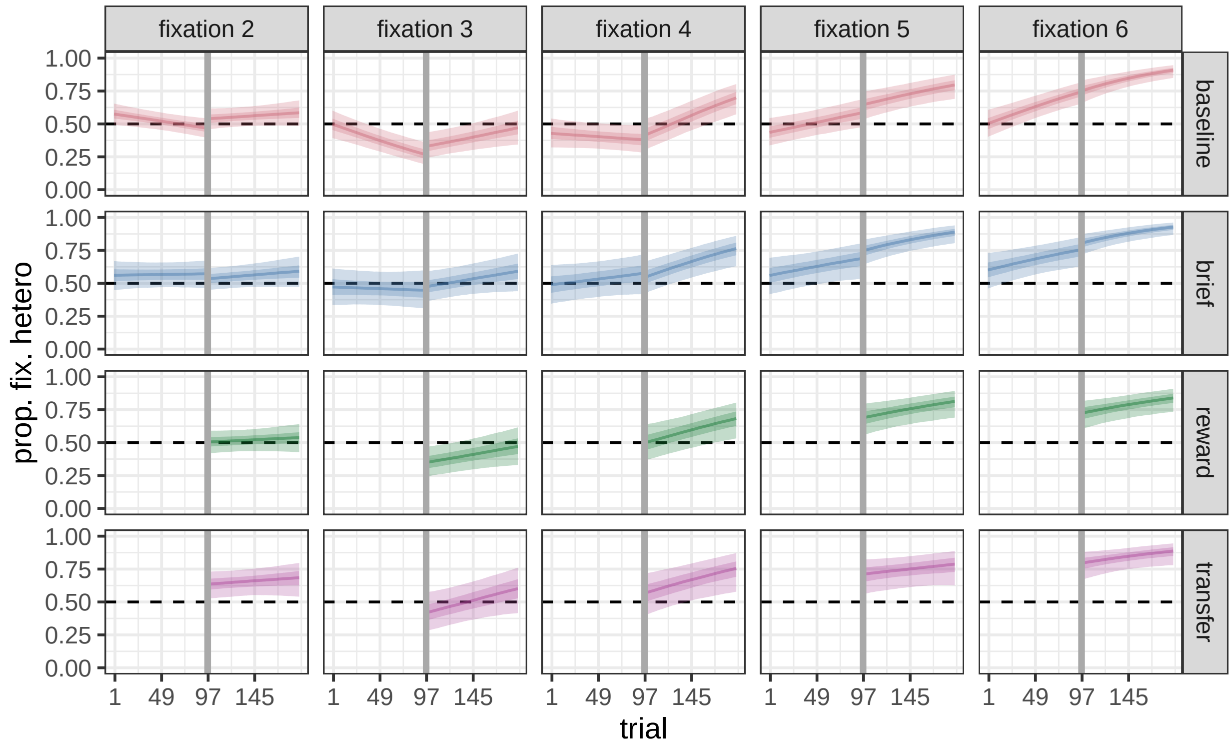
```r
m_posterior <- readRDS("models/my_posterior.model")
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: hetero_fix ~ (0 + cd) * (0 + n) * bk * ts + (ts + n | observer)
##    Data: d_strat (Number of observations: 21830)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Group-Level Effects:
## ~observer (Number of levels: 57)
##                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)          0.93      0.11     0.73     1.16 1.00     1629     2573
## sd(ts)                 0.57      0.07     0.45     0.71 1.00     1310     2590
## sd(n3)                 0.69      0.09     0.54     0.87 1.00     2652     3252
## sd(n4)                 0.90      0.10     0.73     1.12 1.00     1974     2536
## sd(n5)                 0.96      0.11     0.77     1.19 1.00     2139     2849
## sd(n6)                 0.95      0.11     0.75     1.19 1.00     2125     2854
## cor(Intercept,ts)     -0.56      0.11    -0.74    -0.32 1.00     1528     2296
## cor(Intercept,n3)      0.13      0.15    -0.16     0.41 1.00     1822     2750
## cor(ts,n3)             0.21      0.15    -0.10     0.48 1.00     1979     2420
## cor(Intercept,n4)      0.06      0.14    -0.21     0.33 1.00     1566     2990
## cor(ts,n4)             0.21      0.14    -0.07     0.46 1.00     1846     2803
## cor(n3,n4)             0.91      0.04     0.81     0.97 1.00     1560     2759
## cor(Intercept,n5)     -0.10      0.13    -0.36     0.17 1.00     1477     2532
## cor(ts,n5)             0.27      0.13     0.00     0.51 1.00     1784     2655
## cor(n3,n5)             0.79      0.07     0.62     0.90 1.00     1563     2374
## cor(n4,n5)             0.94      0.03     0.87     0.98 1.00     2801     3430
## cor(Intercept,n6)     -0.18      0.13    -0.43     0.09 1.00     1808     2834
## cor(ts,n6)             0.25      0.14    -0.02     0.50 1.00     1913     2775
## cor(n3,n6)             0.64      0.10     0.41     0.81 1.00     1628     2247
## cor(n4,n6)             0.84      0.06     0.71     0.93 1.00     2504     3087
## cor(n5,n6)             0.95      0.03     0.89     0.99 1.00     3080     3481
##
## Population-Level Effects:
##                            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## cdbrief                        0.24      0.28    -0.30     0.77 1.01     1694
## cdbaseline                     0.30      0.20    -0.10     0.70 1.00     1513
## cdtransfer                     0.33      0.77    -1.15     1.87 1.00     6067
## cdreward                       0.30      0.24    -0.17     0.75 1.00     1808
## n3                            -0.36      0.25    -0.86     0.12 1.00     1842
## n4                            -0.29      0.28    -0.82     0.26 1.00     1612
## n5                            -0.00      0.29    -0.56     0.55 1.00     1954
## n6                             0.17      0.29    -0.40     0.74 1.00     2159
## bkblock2                      -0.35      0.37    -1.09     0.38 1.00     2251
## ts                             0.04      0.26    -0.46     0.56 1.00     2104
## cdbaseline:n3                  0.05      0.30    -0.53     0.63 1.00     2012
## cdtransfer:n3                 -0.47      0.79    -2.00     1.07 1.00     6788
## cdreward:n3                    0.10      0.32    -0.54     0.72 1.00     2216
## cdbaseline:n4                 -0.30      0.33    -0.93     0.32 1.00     2216
## cdtransfer:n4                 -0.10      0.77    -1.61     1.40 1.00     7391
## cdreward:n4                    0.45      0.36    -0.25     1.16 1.00     2198
## cdbaseline:n5                 -0.55      0.33    -1.21     0.08 1.00     1994
## cdtransfer:n5                 -0.01      0.82    -1.62     1.62 1.00     8032
```

```
## cdreward:n5                 0.07      0.37    -0.67     0.78 1.00      2326
## cdbaseline:n6              -0.46      0.34    -1.11     0.22 1.00      2440
## cdtransfer:n6              -0.06      0.78    -1.63     1.49 1.00      7596
## cdreward:n6                -0.06      0.37    -0.79     0.65 1.00      2387
## cdbaseline:bkblock2         0.01      0.44    -0.86     0.88 1.00      2470
## cdtransfer:bkblock2         0.35      0.77    -1.17     1.83 1.00      6614
## cdreward:bkblock2          -0.07      0.43    -0.90     0.77 1.00      2703
## n3:bkblock2                -0.12      0.43    -0.98     0.75 1.00      3602
## n4:bkblock2                -0.44      0.44    -1.32     0.40 1.00      4168
## n5:bkblock2                 0.17      0.47    -0.77     1.09 1.00      4231
## n6:bkblock2                 0.22      0.48    -0.71     1.18 1.00      4702
## cdbaseline:ts              -0.47      0.31    -1.10     0.13 1.00      2079
## cdtransfer:ts              -0.00      0.72    -1.40     1.43 1.00      7347
## cdreward:ts                -0.54      0.32    -1.19     0.11 1.00      2266
## n3:ts                      -0.13      0.32    -0.75     0.51 1.00      2988
## n4:ts                       0.31      0.33    -0.35     0.94 1.00      2327
## n5:ts                       0.51      0.33    -0.14     1.18 1.00      2769
## n6:ts                       0.68      0.34    -0.01     1.35 1.00      3773
## bkblock2:ts                 0.20      0.30    -0.37     0.76 1.00      2170
## cdbaseline:n3:bkblock2     -0.84      0.55    -1.92     0.26 1.00      4368
## cdtransfer:n3:bkblock2     -0.47      0.79    -2.02     1.08 1.00      7882
## cdreward:n3:bkblock2       -0.65      0.55    -1.74     0.45 1.00      4609
## cdbaseline:n4:bkblock2     -0.50      0.55    -1.57     0.59 1.00      4858
## cdtransfer:n4:bkblock2     -0.12      0.80    -1.72     1.43 1.00      6820
## cdreward:n4:bkblock2       -0.41      0.58    -1.57     0.72 1.00      5373
## cdbaseline:n5:bkblock2      0.24      0.57    -0.84     1.37 1.00      5202
## cdtransfer:n5:bkblock2     -0.01      0.82    -1.64     1.58 1.00      8824
## cdreward:n5:bkblock2       -0.02      0.61    -1.21     1.17 1.00      5259
## cdbaseline:n6:bkblock2      0.01      0.58    -1.13     1.12 1.00      5040
## cdtransfer:n6:bkblock2     -0.04      0.82    -1.60     1.60 1.00      7520
## cdreward:n6:bkblock2        0.04      0.59    -1.10     1.20 1.00      5129
## cdbaseline:n3:ts           -0.43      0.39    -1.21     0.32 1.00      3442
## cdtransfer:n3:ts            0.15      0.74    -1.32     1.58 1.00      6612
## cdreward:n3:ts             -0.36      0.41    -1.17     0.44 1.00      3508
## cdbaseline:n4:ts           -0.09      0.39    -0.87     0.67 1.00      2889
## cdtransfer:n4:ts           -0.06      0.76    -1.50     1.43 1.00      7109
## cdreward:n4:ts             -0.87      0.41    -1.67    -0.07 1.00      3027
## cdbaseline:n5:ts            0.52      0.40    -0.27     1.31 1.00      2901
## cdtransfer:n5:ts           -0.28      0.75    -1.77     1.15 1.00      6985
## cdreward:n5:ts             -0.12      0.43    -0.95     0.71 1.00      3438
## cdbaseline:n6:ts            0.80      0.41    -0.01     1.58 1.00      3951
## cdtransfer:n6:ts           -0.19      0.78    -1.68     1.35 1.00      7298
## cdreward:n6:ts              0.06      0.43    -0.79     0.91 1.00      4100
## cdbaseline:bkblock2:ts      0.43      0.36    -0.29     1.13 1.00      2451
## cdtransfer:bkblock2:ts     -0.02      0.71    -1.43     1.33 1.00      7551
## cdreward:bkblock2:ts        0.43      0.35    -0.25     1.13 1.00      2512
## n3:bkblock2:ts              0.37      0.37    -0.34     1.09 1.00      3100
## n4:bkblock2:ts              0.46      0.38    -0.26     1.22 1.00      2911
## n5:bkblock2:ts              0.26      0.39    -0.50     1.02 1.00      3268
## n6:bkblock2:ts              0.22      0.40    -0.57     1.02 1.00      3839
## cdbaseline:n3:bkblock2:ts   0.60      0.46    -0.31     1.49 1.00      3575
## cdtransfer:n3:bkblock2:ts   0.14      0.75    -1.33     1.63 1.00      6870
## cdreward:n3:bkblock2:ts     0.51      0.46    -0.40     1.41 1.00      3774
## cdbaseline:n4:bkblock2:ts   0.34      0.46    -0.59     1.23 1.00      3469
```

```
## cdtransfer:n4:bkblock2:ts    -0.06      0.76    -1.52      1.43 1.00     6504
## cdreward:n4:bkblock2:ts       0.76      0.49    -0.20      1.72 1.00     3464
## cdbaseline:n5:bkblock2:ts    -0.70      0.47    -1.65      0.21 1.00     3778
## cdtransfer:n5:bkblock2:ts    -0.30      0.74    -1.76      1.15 1.00     6731
## cdreward:n5:bkblock2:ts      -0.10      0.49    -1.06      0.88 1.00     4355
## cdbaseline:n6:bkblock2:ts    -0.69      0.48    -1.66      0.25 1.00     4011
## cdtransfer:n6:bkblock2:ts    -0.20      0.77    -1.72      1.34 1.00     7046
## cdreward:n6:bkblock2:ts      -0.38      0.50    -1.35      0.59 1.00     4282
##                           Tail_ESS
## cdbrief                       2325
## cdbaseline                    2090
## cdtransfer                    3032
## cdreward                      2543
## n3                            3007
## n4                            2573
## n5                            2745
## n6                            3013
## bkblock2                      2380
## ts                            2675
## cdbaseline:n3                 2398
## cdtransfer:n3                 3030
## cdreward:n3                   2944
## cdbaseline:n4                 2711
## cdtransfer:n4                 3036
## cdreward:n4                   2763
## cdbaseline:n5                 2937
## cdtransfer:n5                 2633
## cdreward:n5                   2418
## cdbaseline:n6                 2904
## cdtransfer:n6                 2770
## cdreward:n6                   2936
## cdbaseline:bkblock2           2720
## cdtransfer:bkblock2           2894
## cdreward:bkblock2             3345
## n3:bkblock2                   3250
## n4:bkblock2                   3404
## n5:bkblock2                   3381
## n6:bkblock2                   3506
## cdbaseline:ts                 2726
## cdtransfer:ts                 2891
## cdreward:ts                   2673
## n3:ts                         3218
## n4:ts                         3048
## n5:ts                         2790
## n6:ts                         3089
## bkblock2:ts                   2868
## cdbaseline:n3:bkblock2        3320
## cdtransfer:n3:bkblock2        3107
## cdreward:n3:bkblock2          3137
## cdbaseline:n4:bkblock2        3030
## cdtransfer:n4:bkblock2        2829
## cdreward:n4:bkblock2          3425
## cdbaseline:n5:bkblock2        3519
## cdtransfer:n5:bkblock2        2654
```

```
## cdreward:n5:bkblock2          3578
## cdbaseline:n6:bkblock2        3486
## cdtransfer:n6:bkblock2        3000
## cdreward:n6:bkblock2          3213
## cdbaseline:n3:ts              3345
## cdtransfer:n3:ts              3482
## cdreward:n3:ts                3348
## cdbaseline:n4:ts              3333
## cdtransfer:n4:ts              2937
## cdreward:n4:ts                3278
## cdbaseline:n5:ts              2636
## cdtransfer:n5:ts              3138
## cdreward:n5:ts                2639
## cdbaseline:n6:ts              2923
## cdtransfer:n6:ts              3123
## cdreward:n6:ts                3459
## cdbaseline:bkblock2:ts        2980
## cdtransfer:bkblock2:ts        2950
## cdreward:bkblock2:ts          3280
## n3:bkblock2:ts                2988
## n4:bkblock2:ts                3107
## n5:bkblock2:ts                3158
## n6:bkblock2:ts                3464
## cdbaseline:n3:bkblock2:ts     3300
## cdtransfer:n3:bkblock2:ts     3219
## cdreward:n3:bkblock2:ts       3479
## cdbaseline:n4:bkblock2:ts     3066
## cdtransfer:n4:bkblock2:ts     2916
## cdreward:n4:bkblock2:ts       3107
## cdbaseline:n5:bkblock2:ts     3252
## cdtransfer:n5:bkblock2:ts     3136
## cdreward:n5:bkblock2:ts       3361
## cdbaseline:n6:bkblock2:ts     3156
## cdtransfer:n6:bkblock2:ts     3259
## cdreward:n6:bkblock2:ts       3213
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).

## Warning: Removed 10220 row(s) containing missing values (geom_path).
```




```
## Warning: Removed 10220 row(s) containing missing values (geom_path).

##    observer                 cd            bk                   t
##  Length:21830       brief    :3509   Length:21830       Min.   :  1.00
##  Class :character   baseline:9378   Class :character   1st Qu.: 49.00
##  Mode  :character   transfer:2026   Mode  :character   Median : 97.00
##                     reward   :6917                      Mean   : 97.23
##                                                         3rd Qu.:145.00
##                                                         Max.   :192.00
##  n           duration        hetero_fix           ts          targ_side
##  2:4545   Min.   :  14.0   Min.   :0.0000   Min.   :0.000   Length:21830
##  3:4486   1st Qu.: 148.0   1st Qu.:0.0000   1st Qu.:0.500   Class :character
##  4:4387   Median : 195.0   Median :1.0000   Median :1.000   Mode  :character
##  5:4258   Mean   : 206.2   Mean   :0.5592   Mean   :1.002
##  6:4154   3rd Qu.: 252.0   3rd Qu.:1.0000   3rd Qu.:1.500
##           Max.   :1185.0   Max.   :1.0000   Max.   :1.990
##        rt              acc
##  Min.   : 0.271   Min.   :1
##  1st Qu.: 2.841   1st Qu.:1
##  Median : 6.401   Median :1
##  Mean   : 9.645   Mean   :1
##  3rd Qu.:13.027   3rd Qu.:1
##  Max.   :53.535   Max.   :1
```

3.3 Can we easily simplify the model?

I now fit simpler models, removing either one of the four variables, or the four-way interaction.

```
## Compiling Stan program...

## Start sampling
```

model	weight	n
m_posterior	0.4398922	80
m_posterior_drop_bk	0.0456466	40
m_posterior_drop_t	0.0000010	40
m_posterior_drop_cd	0.4488511	20
m_posterior_drop_n	0.0656091	16

4 Session Info

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19041)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] RcppRoll_0.3.0  lmerTest_3.1-3  lme4_1.1-26     Matrix_1.2-18
##  [5] patchwork_1.1.1 tidybayes_2.3.1 forcats_0.5.1   stringr_1.4.0
##  [9] dplyr_1.0.4     purrr_0.3.4     readr_1.4.0     tidyr_1.1.2
## [13] tibble_3.0.6    ggplot2_3.3.3   tidyverse_1.3.0 brms_2.14.4
## [17] Rcpp_1.0.6
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1        backports_1.2.1     plyr_1.8.6
##   [4] igraph_1.2.6        splines_4.0.3       svUnit_1.0.3
##   [7] crosstalk_1.1.1     rstantools_2.1.1    inline_0.3.17
##  [10] digest_0.6.27       htmltools_0.5.1.1   rsconnect_0.8.16
##  [13] fansi_0.4.2         checkmate_2.0.0     magrittr_2.0.1
##  [16] modelr_0.1.8        RcppParallel_5.0.2  matrixStats_0.58.0
##  [19] xts_0.12.1          prettyunits_1.1.1   colorspace_2.0-0
##  [22] rvest_0.3.6         ggdist_2.4.0        haven_2.3.1
##  [25] xfun_0.20           callr_3.5.1         crayon_1.4.0
##  [28] jsonlite_1.7.2      zoo_1.8-8           glue_1.4.2
##  [31] gtable_0.3.0        V8_3.4.0            distributional_0.2.2
##  [34] pkgbuild_1.2.0      rstan_2.21.2        abind_1.4-5
##  [37] scales_1.1.1        mvtnorm_1.1-1       DBI_1.1.1
##  [40] ggthemes_4.2.4      miniUI_0.1.1.1      xtable_1.8-4
##  [43] stats4_4.0.3        StanHeaders_2.21.0-7 DT_0.17
##  [46] htmlwidgets_1.5.3   httr_1.4.2          threejs_0.3.3
##  [49] arrayhelpers_1.1-0  ellipsis_0.3.1      pkgconfig_2.0.3
##  [52] loo_2.4.1           farver_2.0.3        dbplyr_2.1.0
##  [55] utf8_1.1.4          tidyselect_1.1.0    labeling_0.4.2
##  [58] rlang_0.4.10        reshape2_1.4.4      later_1.1.0.1
##  [61] munsell_0.5.0       cellranger_1.1.0    tools_4.0.3
```

```
##  [64] cli_2.3.0           generics_0.1.0      broom_0.7.4
##  [67] ggridges_0.5.3      evaluate_0.14       fastmap_1.1.0
##  [70] yaml_2.2.1          processx_3.4.5      knitr_1.31
##  [73] fs_1.5.0            nlme_3.1-149        mime_0.9
##  [76] projpred_2.0.2      xml2_1.3.2          compiler_4.0.3
##  [79] bayesplot_1.8.0     shinythemes_1.2.0   rstudioapi_0.13
##  [82] curl_4.3            gamm4_0.2-6         reprex_1.0.0
##  [85] statmod_1.4.35      stringi_1.5.3       ps_1.5.0
##  [88] Brobdingnag_1.2-6   lattice_0.20-41     nloptr_1.2.2.2
##  [91] markdown_1.1        shinyjs_2.0.0       vctrs_0.3.6
##  [94] pillar_1.4.7        lifecycle_0.2.0     bridgesampling_1.0-0
##  [97] httpuv_1.5.5        R6_2.5.0            bookdown_0.21
## [100] promises_1.1.1      gridExtra_2.3       codetools_0.2-16
## [103] boot_1.3-25         colourpicker_1.1.0  MASS_7.3-53
## [106] gtools_3.8.2        assertthat_0.2.1    withr_2.4.1
## [109] shinystan_2.5.0     mgcv_1.8-33         parallel_4.0.3
## [112] hms_1.0.0           grid_4.0.3          coda_0.19-4
## [115] minqa_1.2.4         rmarkdown_2.6       numDeriv_2016.8-1.1
## [118] shiny_1.6.0         lubridate_1.7.9.2   base64enc_0.1-3
## [121] dygraphs_1.1.1.6
```