Multiple Linear Regression and Model Comparison

PSY517 Quantitative Analysis III

Derek Powell

Module 2

2021-09-14

Multiple Linear Regression and Model Comparison
PSY517 Quantitative Analysis III
Derek Powell
Module 2

Assessing model fit

Once we fit a model, we want to know: how good is our model?

Some common measures:

- Variance accounted for: R^2
- Probability
    - log-likelihood (aka deviance)
    - posterior probability
- Cross validation
- Information criteria

Assessing model fit

Once we fit a model, we want to know: how good is our model?

Some common measures:

- Variance accounted for: R^2
- Probability
    - log-likelihood (aka deviance)
    - posterior probability
- Cross validation
- Information criteria

20m on model comparison theory (10 slides max) 5m on feature integration theory / visual search (3 slides) 40m on multiple regression + application of model comparison (30 slides max) 10m on class business

No alarms and no surprises, please

- A good model means we should not be surprised by new observations.
- **Information theory** gives us a measure of *surprising-ness*

$$H(p) = -E[log(p_i)]$$

- Can estimate how surprising the observed data are given our model as the log-pointwise-predictive-density:

$$\widehat{\text{lppd}} = \sum_i^N log\left(\frac{1}{S}\sum_s p(y_i|\theta_s)\right)$$

No alarms and no surprises, please

No alarms and no surprises, please

- A good model means we should not be surprised by new observations.
- **Information theory** gives us a measure of *surprising-ness*

$$H(p) = -E[log(p_i)]$$

- Can estimate how surprising the observed data are given our model as the log-pointwise-predictive-density:

$$\widehat{\text{lppd}} = \sum_i^N log\left(\frac{1}{S}\sum_s p(y_i|\theta_s)\right)$$

Generalization

As scientists, we don't just want to describe what is happening in our sample of observations, but also to generalize to new unseen data.

So, we don't just want to know the lppd, we want elppd: the expected log posterior predictive density for new data $\tilde{y}$

$$\text{elppd} = \sum_{i}^{N} \int \log(\, p(\tilde{y}_i|y))\, p_t(\tilde{y}_i) d\tilde{y}$$

Where $p_t(\tilde{y}_i)$ is the true probability of new data $\tilde{y}_i$, which we can't know.

Generalization

As scientists, we don't just want to describe what is happening in our sample of observations, but also to generalize to new unseen data.

So, we don't just want to know the lppd, we want elppd: the expected log posterior predictive density for new data $\tilde{y}$

$$\text{elppd} = \sum_{i}^{N} \int \log(\, p(\tilde{y}_i|y))\, p_t(\tilde{y}_i) d\tilde{y}$$

Where $p_t(\tilde{y}_i)$ is the true probability of new data $\tilde{y}_i$, which we can't know.

Don't worry about this equation too much, point is just that we are missing a piece needed to calculate what we want

Leave-one-out cross validation

We can estimate the model's generalization performance using **cross validation**.

$$\widehat{\text{elppd}} = \sum_{i}^{N} log\, P(y_i | y_{-i})$$

- For n observations we create n datasets each with one observation held out
- n models are fit to each of these n datasets and each time used to predict the held-out observation's value

Schematically:

```
for (i in 1:nrow(df)){
  d <- df[-i, ]
  m <- lm(y ~ x, data = d)
  pred <- predict(m, newdata = df[i, ])
  compute_error(pred, df$y[i])
}
```


Leave-one-out cross validation

We can estimate the model's generalization performance using **cross validation**.

$$\widehat{\text{elppd}} = \sum_{i}^{N} log\, P(y_i | y_{-i})$$

- For n observations we create n datasets each with one observation held out
- n models are fit to each of these n datasets and each time used to predict the held-out observation's value

Schematically:

```
for (i in 1:nrow(df)){
  d <- df[-i, ]
  m <- lm(y ~ x, data = d)
  pred <- predict(m, newdata = df[i, ])
  compute_error(pred, df$y[i])
}
```


- earlier said we want to know how surprised we are likely to be by new data
- can estimate that with cross validation

Variance accounted for R^2

$$R^2 = \frac{Var(\text{outcome}) - Var(\text{residuals})}{Var(\text{outcome})}$$

- R^2 is the "proportion of variance explained" by a model.
- It is an **absolute** measure of model fit that ranges from 0 (no fit at all) to 1 (perfect fit)
- R^2 is very useful, but it is also not to be trusted
- It does nothing to account for **model complexity**

Variance accounted for R^2

$$R^2 = \frac{Var(\text{outcome}) - Var(\text{residuals})}{Var(\text{outcome})}$$

- R^2 is the "proportion of variance explained" by a model.
- It is an **absolute** measure of model fit that ranges from 0 (no fit at all) to 1 (perfect fit)
- R^2 is very useful, but it is also not to be trusted
- It does nothing to account for **model complexity**

- ok that's where we're going
- let's turn to something more familiar: R2
- why are we going to be interested in cross validation? Why not just use R2?

Model complexity

- Model complexity refers to how flexible the model is to accommodate different patterns of data
- More complex models are more flexible and can fit more different patterns of data
- Adding more predictors will always increase R^2, even if they are not meaningful
- E.g. the model below can perfectly fit any n points:

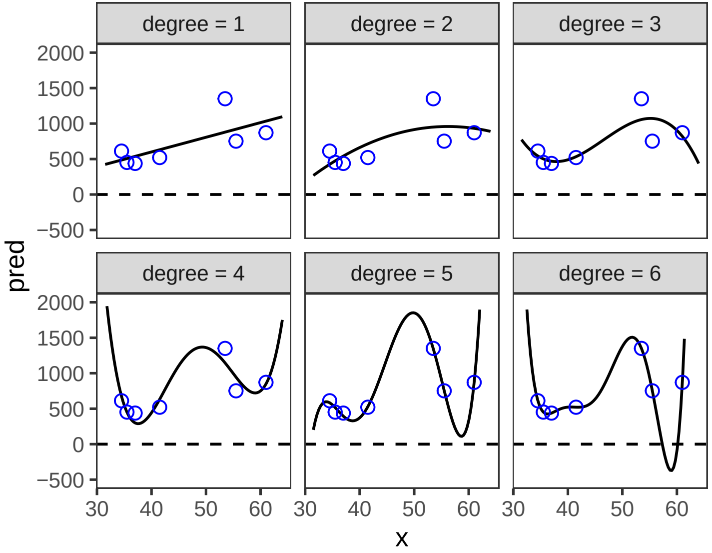
$$\mu_i = \beta_1 x + \beta_2 x^2 + \ldots + \beta_{n-1} x^{n-1}$$

Model complexity

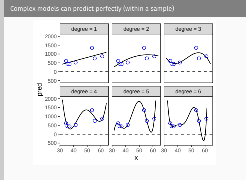
- Model complexity refers to how flexible the model is to accommodate different patterns of data
- More complex models are more flexible and can fit more different patterns of data
- Adding more predictors will always increase R^2, even if they are not meaningful
- E.g. the model below can perfectly fit any n points:

$$\mu_i = \beta_1 x + \beta_2 x^2 + \ldots + \beta_{n-1} x^{n-1}$$

Complex models can predict perfectly (within a sample)

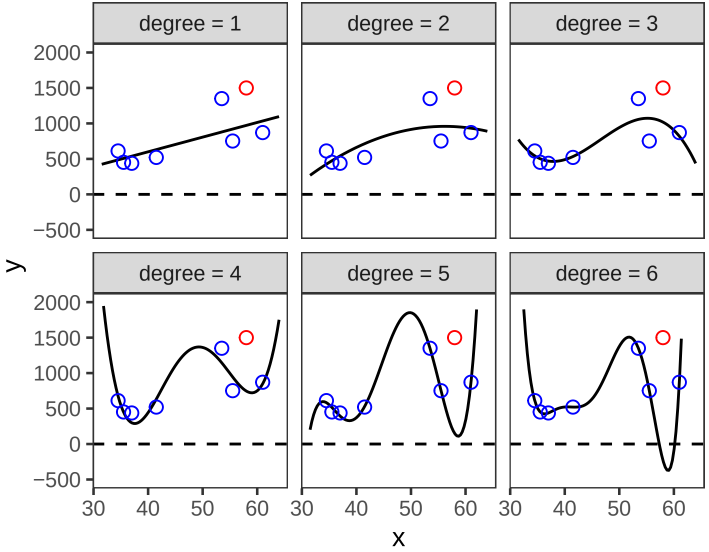


└─Complex models can predict perfectly (within a sample)



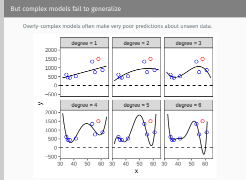
But complex models fail to generalize

Overly-complex models often make very poor predictions about unseen data.



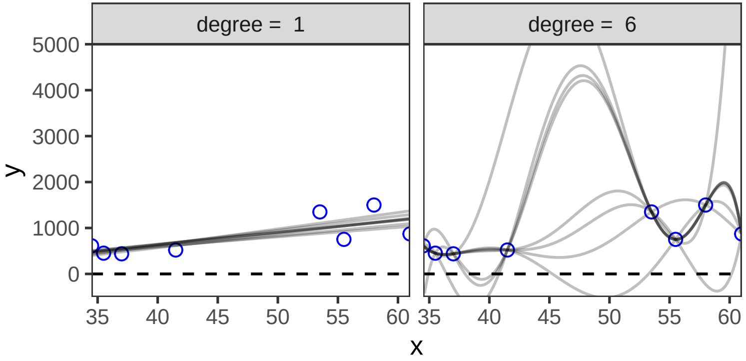
2021-09-14

But complex models fail to generalize



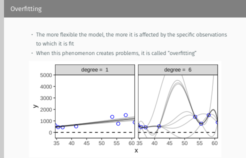
Overfitting

- The more flexible the model, the more it is affected by the specific observations to which it is fit
- When this phenomenon creates problems, it is called “overfitting”



2021-09-14

└Overfitting



- The plot below shows how the regression lines for the simple linear model and the 6-degree polynomial model.
- there are 7 regression lines, each fit holding out one observation
- the simple model is hardly affected at all, but the complex model is wildly different
- occam’s razor
- “entities should not be multiplied beyond necessity”
- received as the simplest explanation is usually the best one.”
- this is one reason to believe this principle / use the razor

- **Information Criteria** are a kind of model fit measure that approximate out-of-sample predictive accuracy
- Some examples:
  - AIC
  - WAIC
  - LOOIC

Information criteria

- Information Criteria are a kind of model fit measure that approximate out-of-sample predictive accuracy
- Some examples:
  - AIC
  - WAIC
  - LOOIC

- We will generally use LOOIC, but we'll start with an explanation of *AIC* and *WAIC* which are a bit simpler to describe.

Akaike Information Criteria

- The ***Akaike Information Criteria*** (*AIC*) is the simplest approximations of leave-one-out cross validation

$$\widehat{\text{elpd}}_{\text{AIC}} = log\, p(y|\hat{\theta}_{mle}) - k$$

- To turn this into the information criteria statistic, it is conventional to multiply this by -2.

$$\text{AIC} = -2(log\, p(y|\hat{\theta}_{mle}) - k)$$

- For information criteria, **lower numbers indicate better model fit.**

Akaike Information Criteria

- The ***Akaike Information Criteria*** (AIC) is the simplest approximations of leave-one-out cross validation

$$\widehat{\text{elpd}}_{\text{AIC}} = log\, p(y|\hat{\theta}_{mle}) - k$$

- To turn this into the information criteria statistic, it is conventional to multiply this by -2.

$$\text{AIC} = -2(log\, p(y|\hat{\theta}_{mle}) - k)$$

- For information criteria, **lower numbers indicate better model fit.**

- based on the maximum likelihood estimate of a model's parameters
- k is number of parameters

Leave-One-Out Information Criteria (LOOIC)

- *LOOIC* is a newer better approximation to leave-one-out cross validation
- The **brms** function `loo()` uses some clever math to directly estimate $\widehat{\text{elppd}}_{\text{loo}}$ without actually having to re-fit the model for each of the n data points.
- In this class we will usually work with $\widehat{\text{elppd}}_{\text{loo}}$ rather than *LOOIC* (which is multiplied by -2)
- Remember: Everything that depends on parameters has uncertainty, and this includes $\widehat{\text{elppd}}_{\text{loo}}$ and *LOOIC*

Leave-One-Out Information Criteria (LOOIC)

- *LOOIC* is a newer better approximation to leave-one-out cross validation
- The **brms** function `loo()` uses some clever math to directly estimate $\widehat{\text{elppd}}_{\text{loo}}$ without actually having to re-fit the model for each of the n data points.
- In this class we will usually work with $\widehat{\text{elppd}}_{\text{loo}}$ rather than *LOOIC* (which is multiplied by -2)
- Remember: Everything that depends on parameters has uncertainty, and this includes $\widehat{\text{elppd}}_{\text{loo}}$ and *LOOIC*

- another reason to prefer over R2: elppd is THE THING that we want, for all models

- `loo()` will also estimate the "effective number of parameters" $\hat{p}_{\text{loo}}$ as

$$\hat{p}_{\text{loo}} = \widehat{\text{lppd}} - \widehat{\text{elppd}}_{\text{loo}}$$

- More complex models will always have as high or higher $\widehat{\text{lppd}}$ compared with simpler models, but they will often perform worse on new data.
- Therefore, the difference between $\widehat{\text{lppd}}$ and $\widehat{\text{elppd}}$ is a measure of model complexity

Estimating model complexity with `loo()`

- `loo()` will also estimate the "effective number of parameters" $\hat{p}_{\text{loo}}$ as

$$\hat{p}_{\text{loo}} = \widehat{\text{lppd}} - \widehat{\text{elppd}}_{\text{loo}}$$

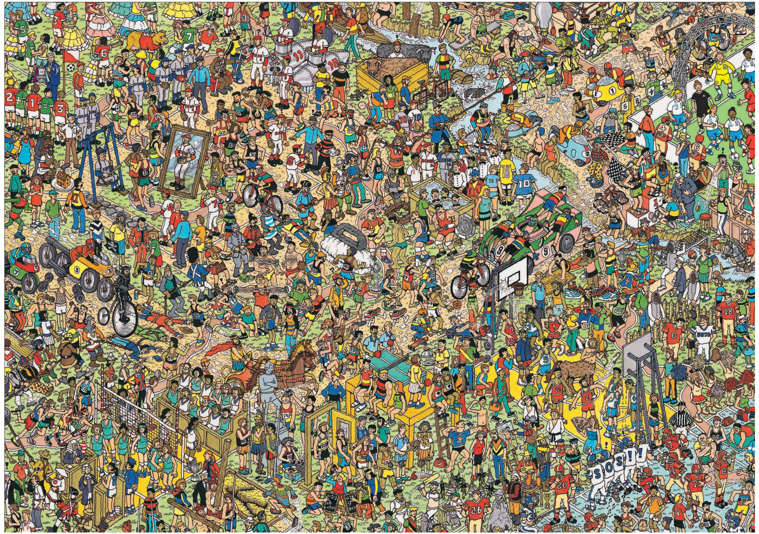
- More complex models will always have as high or higher $\widehat{\text{lppd}}$ compared with simpler models, but they will often perform worse on new data.
- Therefore, the difference between $\widehat{\text{lppd}}$ and $\widehat{\text{elppd}}$ is a measure of model complexity

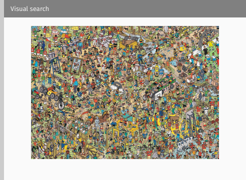
When the difference (elpd_diff) is larger than 4, the number of observations is larger than 100 and the model is not badly misspecified then normal approximation and SE are quite reliable description of the uncertainty in the difference. Differences smaller than 4 are small and then the models have very similar predictive performance and it doesn't matter if the normal approximation fails or SE is underestimated. - per: https://avehtari.github.io/modelselection/CV-FAQ.html#14_Why_(sqrt%7Bn%7D)_in_Standard_error_(SE)_of_LOO

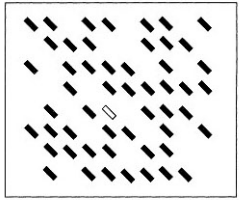
Multiple regression example

Multiple regression example

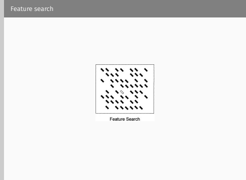
Visual search



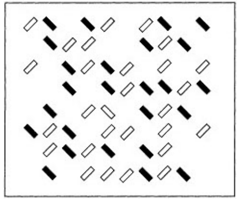




Feature Search

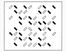


Conjunction search



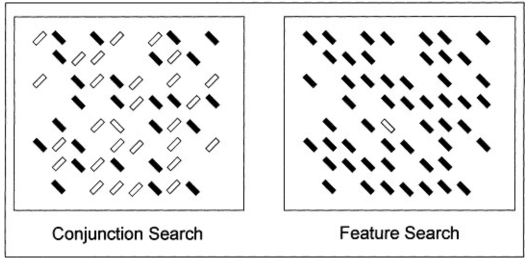
Conjunction Search

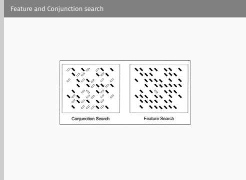
Conjunction search



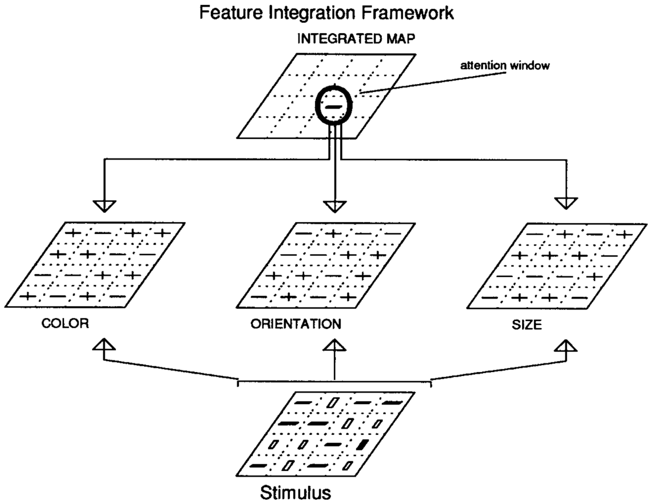
Conjunction Search

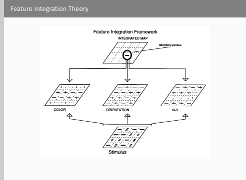
Feature and Conjunction search





Feature Integration Theory





Visual search data

- 18 participants randomly assigned to feature or conjunction search conditions (between subjects)
- Completed with 3, 6, 12, and 18 total items in the display
- Response times measured and averaged from approximately 400 trials for each condition
- First focusing on trials where target was present in the display
- Data collected by Jeremy Wolfe

```
df1 <- vizsearch %>%
  filter(targ_absent==0)
```


Visual search data
- 18 participants randomly assigned to feature or conjunction search conditions (between subjects)
- Completed with 3, 6, 12, and 18 total items in the display
- Response times measured and averaged from approximately 400 trials for each condition
- First focusing on trials where target was present in the display
- Data collected by Jeremy Wolfe
df1 <- vizsearch %>%
  filter(targ_absent==0)

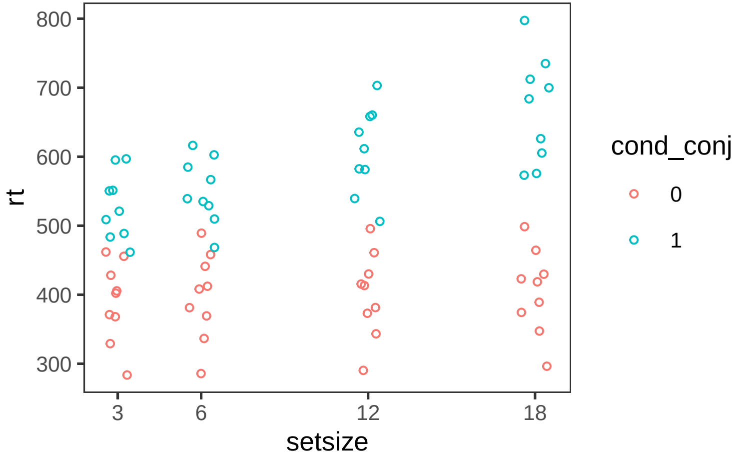
Looking at the data

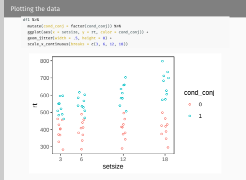
```
head(df1)
```

```
## # A tibble: 6 x 6
##   subject cond_conj targ_absent setsize setsize_s    rt
##   <chr>       <dbl>       <dbl>   <dbl>     <dbl> <dbl>
## 1 ag              0           0       3         0  456.
## 2 ag              0           0       6         3  458.
## 3 ag              0           0      12         9  461.
## 4 ag              0           0      18        15  464.
## 5 ak              0           0       3         0  368.
## 6 ak              0           0       6         3  369.
```

Plotting the data

```
df1 %>%
  mutate(cond_conj = factor(cond_conj)) %>%
  ggplot(aes(x = setsize, y = rt, color = cond_conj)) +
  geom_jitter(width = .5, height = 0) +
  scale_x_continuous(breaks = c(3, 6, 12, 18))
```





Proposing the model: likelihood

$$\text{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 C_i + \beta_2 S_i + \beta_3\, C_i S_i$$

Remember: Each coefficient represents the effect on the outcome of a one-unit change in the predictor when all other predictors are equal to zero.

Proposing the model: likelihood

$$\text{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 C_i + \beta_2 S_i + \beta_3\, C_i S_i$$

Remember: Each coefficient represents the effect on the outcome of a one-unit change in the predictor when all other predictors are equal to zero.

- Because the setsize is never actually zero, I could shift the `setsize` variable so that this zero value in the model will have a meaningful interpretation.
- But I don't care terrible about the intercept or other coefficient
- Centering variables is another popular solution. But it is always nice to think about what is meaningful for your data.

Proposing the model: priors

To scale my weakly-informative priors, I first calculated the **mean** and **sd** of the data:

```r
mean(df1$rt)
```

```
## [1] 493.4627
```

```r
sd(df1$rt)
```

```
## [1] 118.4061
```

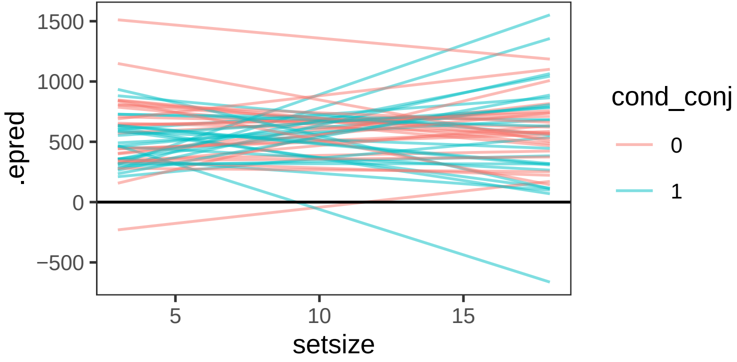
$$\text{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 C_i + \beta_2 S_i + \beta_3\, C_i S_i$$
$$\alpha \sim Normal(500, 120)$$
$$\beta_1 \sim Normal(0, 360)$$
$$\beta_2 \sim Normal(0, 24)$$
$$\beta_3 \sim Normal(0, 24)$$
$$\sigma \sim Exponential(1/200)$$

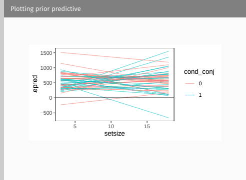
```r
fit1 <- brm(
  rt ~ setsize + cond_conj  + setsize:cond_conj,
  prior =
    prior(normal(500, 120), class=Intercept) +
    prior(normal(0, 360), coef=cond_conj) +
    prior(normal(0, 24), coef=setsize) +
    prior(normal(0, 24), coef=`setsize:cond_conj`) +
    prior(exponential(.005), class=sigma),
  family = gaussian(),
  data = df1
  )
```


brms auto-centered intercept at grand mean ~500. I let that vary by 1 sd b/c I know that puts things in a reasonble range, people can't be faster than about 250ms or 2sd different

- For other coefficients I used even less info, but made sure to respect the scaling of the variables.
- Used 3x standard deviation. This is very loose and allows for effects we essentially never see in pscyhology, but rules out e.g. 1 million, etc.
- `cond_conj` ranges from 0 and 1
- `setsize_s` ranges from 0 to 15 (3 to 18 originally), so need to scale by 15.

Plotting prior predictive





Fitting the model

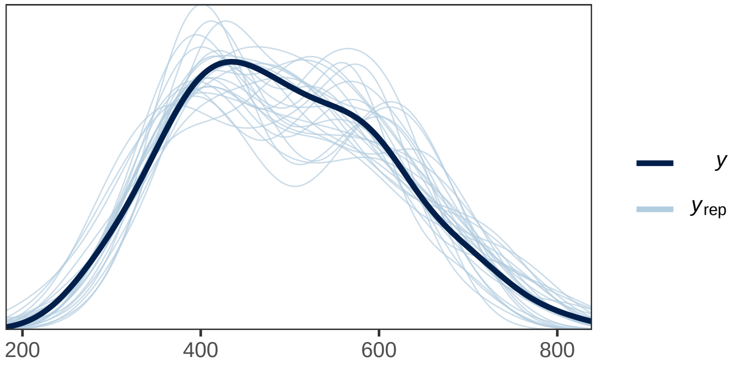
```r
summary(fit1)
```

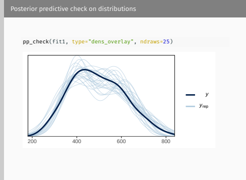
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: rt ~ setsize + cond_conj + setsize:cond_conj
##    Data: df1 (Number of observations: 72)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept           389.76     19.42   352.65   428.37 1.00     2256     2446
## setsize               0.87      1.71    -2.63     4.08 1.00     2060     2310
## cond_conj           107.38     27.61    51.26   161.80 1.00     1921     2042
## setsize:cond_conj     8.54      2.44     3.93    13.39 1.00     1804     2272
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    59.71      5.16    50.48    70.61 1.00     2710     2489
##
## Draws were sampled 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).
```


Fitting the model
summary(fit1)

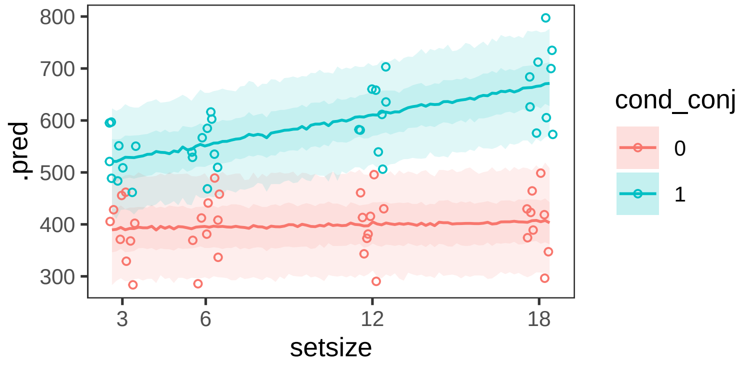
Posterior predictive check on distributions

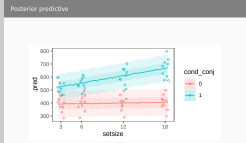
```
pp_check(fit1, type="dens_overlay", ndraws=25)
```





Posterior predictive

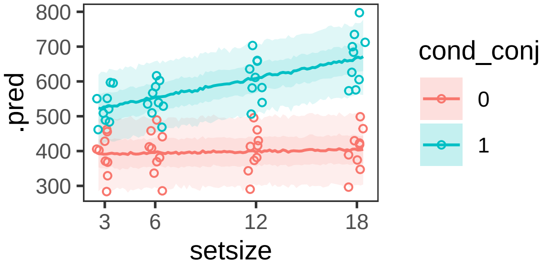


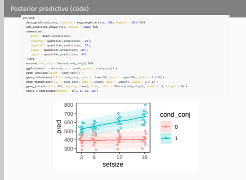


- Used 50% and 90% credible interval.
- I like to use an interval that helps for visual diagnostics. can be good to use multiple intervals too

Posterior predictive (code)

```r
df1 %>%
  data_grid(cond_conj, setsize = seq_range(setsize, 100, expand = .05)) %>%
  add_predicted_draws(fit1, ndraws = 1000) %>%
  summarize(
    .pred = mean(.prediction),
    .lower50 = quantile(.prediction, .25),
    .upper50 = quantile(.prediction, .75),
    .lower = quantile(.prediction, .05),
    .upper = quantile(.prediction, .95)
  ) %>%
  mutate(cond_conj = factor(cond_conj)) %>%
  ggplot(aes(x = setsize, y = .pred, group = cond_conj)) +
  geom_line(aes(color = cond_conj)) +
  geom_ribbon(aes(fill = cond_conj, ymin = .lower50, ymax = .upper50), alpha = 1 / 8) +
  geom_ribbon(aes(fill = cond_conj, ymin = .lower, ymax = .upper), alpha = 1 / 8) +
  geom_jitter(data = df1, mapping = aes(y = rt, color = factor(cond_conj)), width = .5, height = 0) +
  scale_x_continuous(breaks = c(3, 6, 12, 18))
```





Reduced model 1

$$\mathrm{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 C_i + \beta_2 S_i$$
$$\alpha \sim Normal(500, 120)$$
$$\beta_1 \sim Normal(0, 360)$$
$$\beta_2 \sim Normal(0, 24)$$
$$\sigma \sim Exponential(1/200)$$

Reduced model 2

$$\mathrm{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 S_i$$
$$\alpha \sim Normal(500, 120)$$
$$\beta_1 \sim Normal(0, 24)$$
$$\sigma \sim Exponential(1/200)$$

Reduced model 3

$$\mathrm{rt}_i \overset{iid}{\sim} Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \beta_1 C_i$$
$$\alpha \sim Normal(500, 120)$$
$$\beta_1 \sim Normal(0, 360)$$
$$\sigma \sim Exponential(1/200)$$

“Reduced” models
Reduced model 1 Reduced model 2 Reduced model 3

Using information criteria

- IC are **relative** measures of model fit meant for comparing models against one another
- Comparisons between models must be based on the same data y
- Lower is better for information criteria like *AIC*
- Higher is better for $\widehat{\text{elppd}}$ estimates
- These fit indices come with **uncertainty**:
  - Difference between models should be greater than 2 SE to be considered meaninful (some say greater than 4SE)
  - Rule of thumb: differences smaller than 4 are "small" and likely not very consequential

Using information criteria

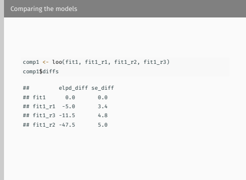
- IC are **relative** measures of model fit meant for comparing models against one another
- Comparisons between models must be based on the same data y
- Lower is better for information criteria like AIC
- Higher is better for $\widehat{\text{elppd}}$ estimates
- These fit indices come with **uncertainty**:
  - Difference between models should be greater than 2 SE to be considered meaninful (some say greater than 4SE)
  - Rule of thumb: differences smaller than 4 are "small" and likely not very consequential

here we're dealing with a normal model, so IC error estimates should be well-behaved

Comparing the models

```r
comp1 <- loo(fit1, fit1_r1, fit1_r2, fit1_r3)
comp1$diffs
```

```
##         elpd_diff se_diff
## fit1      0.0       0.0
## fit1_r1  -5.0       3.4
## fit1_r3 -11.5       4.8
## fit1_r2 -47.5       5.0
```

- interaction term is positive
- visual inspection suggests the interaction model is better but
- but model comparison not conclusive.
- highlights importance of estimates being "conditional on the model" –if we know this model is right, then the parameter is non-zero. But we don't know the model is right or preferred, maybe there's an equally good model out there

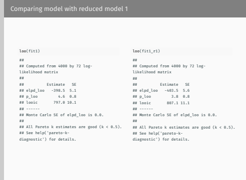
Comparing model with reduced model 1

```
loo(fit1)
```

```
##
## Computed from 4000 by 72 log-
likelihood matrix
##
##          Estimate   SE
## elpd_loo   -398.5  5.1
## p_loo         4.6  0.8
## looic       797.0 10.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-
diagnostic') for details.
```

```
loo(fit1_r1)
```

```
##
## Computed from 4000 by 72 log-
likelihood matrix
##
##          Estimate   SE
## elpd_loo   -403.5  5.6
## p_loo         3.8  0.8
## looic       807.1 11.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-
diagnostic') for details.
```

- like MCMC, loo has diagnostics called pareto_k values
- pareto_k > .70 is bad but a few could be ok.
- pareto_k > 1.0 is definitely bad and result cannot be trusted

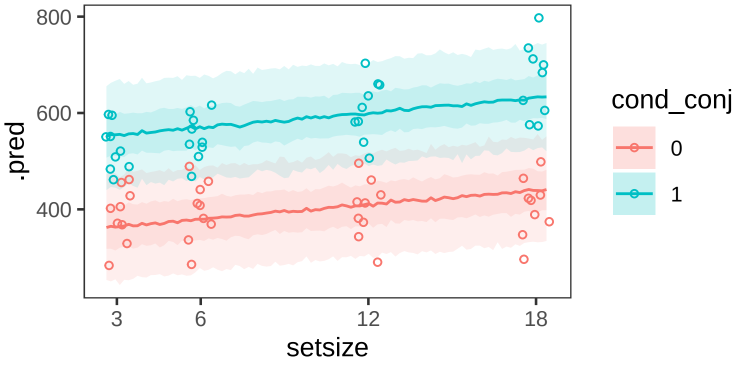
Like p-values, parameter distributions are conditional on the model

```
fit1_lm <- lm(rt ~ setsize + cond_conj + setsize:cond_conj, data = df1)
summary(fit1_lm)
```

```
##
## Call:
## lm(formula = rt ~ setsize + cond_conj + setsize:cond_conj, data = df1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -109.75  -37.60   11.59   46.11  131.17
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       389.5015    19.2553  20.228  < 2e-16 ***
## setsize             0.8849     1.7003   0.520 0.604428
## cond_conj         107.7820    27.2310   3.958 0.000183 ***
## setsize:cond_conj   8.5009     2.4046   3.535 0.000738 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 58.77 on 68 degrees of freedom
## Multiple R-squared:  0.764,  Adjusted R-squared:  0.7536
## F-statistic:  73.4 on 3 and 68 DF,  p-value: < 2.2e-16
```


- p-values and posterior probabilities both suggest that the coefficient is non-zero
- But those are small-world probabilities: true only in the model
- $\widehat{\text{elppd}}_{\text{loo}}$ is warning us that a simpler model might also be plausible

Visual inspection of reduced model 1



- Can see that it's not as good, but the differences are fairly subtle

Adding target-absent trials

- Next I'll make a new dataset **df2** that includes both the target-present and target-absent trials.

```
df2 <- vizsearch
```


Adding target-absent trials
Next I'll make a new dataset df2 that includes both the target-present and target-absent trials.
df2 <- vizsearch

- Analyzing more data might help us gain more confident about our model

Comparing with reduced model 1

```
fit2 <- update(fit1, newdata = df2)
fit2_r1 <- update(fit1_r1, newdata = df2)
```

```
comp2 <- loo(fit2, fit2_r1)
comp2$diffs
```

```
##         elpd_diff se_diff
## fit2       0.0       0.0
## fit2_r1 -10.6       3.8
```


Comparing with reduced model 1

fit2 <- update(fit1, newdata = df2)
fit2_r1 <- update(fit1_r1, newdata = df2)
comp2 <- loo(fit2, fit2_r1)
comp2$diffs
elpd_diff se_diff
fit2   0.0   0.0
fit2_r1 -10.6   3.8

- more data makes us more confident the more complex model is better and more willing to interpret the coefficients

Comparing fits with and without target-present trials

```
bayes_R2(fit1)
```

```
##      Estimate  Est.Error      Q2.5     Q97.5
## R2 0.7561187 0.02660413 0.6920403 0.7945381
```

```
bayes_R2(fit2)
```

```
##      Estimate  Est.Error      Q2.5     Q97.5
## R2 0.6162145 0.03108861 0.5475055 0.6672534
```

```
fixef(fit1)
```

```
##                      Estimate Est.Error       Q2.5      Q97.5
## Intercept         389.7557153 19.420586 352.649968 428.366229
## setsize             0.8668924  1.712280  -2.625593   4.083026
## cond_conj         107.3785052 27.614018  51.262835 161.804456
## setsize:cond_conj   8.5409274  2.444685   3.931355  13.393897
```

```
fixef(fit2)
```

```
##                      Estimate Est.Error       Q2.5      Q97.5
## Intercept         405.5317564 28.250752 348.912669 459.971197
## setsize             0.2616652  2.476358  -4.465968   5.163971
## cond_conj          91.8224372 39.241125  15.225524 168.122474
## setsize:cond_conj  17.2328706  3.456195  10.386644  23.870241
```


But compare to fit1: - compare coefficients, they've changed - R^2 is worse - posterior predictive check looks worse

Enriching the model

- We think there are two processes:
  - **Feature search** happens in parallel for all items at once
  - **Conjunction search** happens serially, item-by-item

How would serial search differ when the target is or is not present in the display?

Enriching the model

- We think there are two processes:
  - Feature search happens in parallel for all items at once
  - Conjunction search happens serially, item-by-item

How would serial search differ when the target is or is not present in the display?

Exhaustive vs. self-terminating serial search

- In our model we treated target-absent and target-present trials exactly the same
- This could represent **exhaustive serial search**: A search process that proceeds item-by-item but only stops once all the items have been searched.
- **self-terminating serial search**: search stops (terminates) once the target is found
    - **Target-present**: with n items in random locations we expect $n/2$ searches are required
    - **Target-absent**: with n items n searches are required

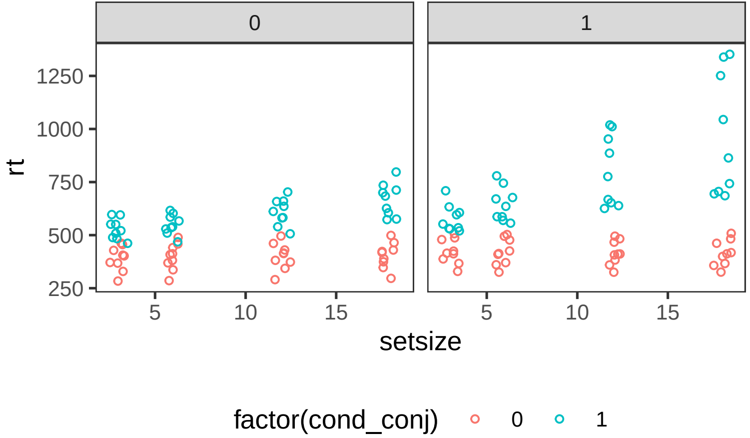
Exhaustive vs. self-terminating serial search

- In our model we treated target-absent and target-present trials exactly the same
- This could represent **exhaustive serial search**: A search process that proceeds item-by-item but only stops once all the items have been searched.
- **self-terminating serial search**: search stops (terminates) once the target is found
    - **Target-present**: with n items in random locations we expect $n/2$ searches are required
    - **Target-absent**: with n items n searches are required

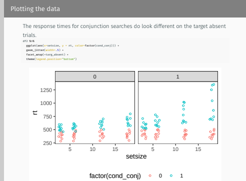
Plotting the data

The response times for conjunction searches do look different on the target absent trials.

```
df2 %>%
  ggplot(aes(x=setsize, y = rt, color=factor(cond_conj))) +
  geom_jitter(width=.5) +
  facet_wrap(~targ_absent) +
  theme(legend.position="bottom")
```



2021-09-14



A model of self-terminating serial search

If search is self-terminating, then the influence of setsize (S) depends on whether the target is present or absent (T), which we capture by adding a 3-way interaction term to the model.

$$
\begin{aligned}
\text{rt}_i &\overset{iid}{\sim} Normal(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta_1 C_i + \beta_2 S_i + \beta_3 C_i S_i + \beta_4 C_i S_i T_i \\
\alpha &\sim Normal(500, 120) \\
\beta_1 &\sim Normal(0, 360) \\
\beta_2 &\sim Normal(0, 24) \\
\beta_3 &\sim Normal(0, 24) \\
\beta_4 &\sim Normal(0, 24) \\
\sigma &\sim Exponential(1/200)
\end{aligned}
$$

A model of self-terminating serial search

If search is self-terminating, then the influence of setsize (S) depends on whether the target is present or absent (T), which we capture by adding a 3-way interaction term to the model.

$$
\begin{aligned}
\text{rt}_i &\overset{iid}{\sim} Normal(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta_1 C_i + \beta_2 S_i + \beta_3 C_i S_i + \beta_4 C_i S_i T_i \\
\alpha &\sim Normal(500, 120) \\
\beta_1 &\sim Normal(0, 360) \\
\beta_2 &\sim Normal(0, 24) \\
\beta_3 &\sim Normal(0, 24) \\
\beta_4 &\sim Normal(0, 24) \\
\sigma &\sim Exponential(1/200)
\end{aligned}
$$

Fitting the larger model

```r
fit3 <- brm(
  rt ~ setsize + cond_conj + setsize:cond_conj + setsize:cond_conj:targ_absent,
  prior = prior(normal(500, 120), class="Intercept") +
    prior(normal(0, 24), coef="setsize") +
    prior(normal(0, 360), coef="cond_conj") +
    prior(normal(0, 24), coef="setsize:cond_conj") +
    prior(normal(0, 24), coef="setsize:cond_conj:targ_absent") +
    prior(exponential(.005), class="sigma"),
  data = df2
)
```

Comparing the models

```r
comp3 <- loo(fit3, fit2)
comp3$diffs
```

```
##      elpd_diff se_diff
## fit3   0.0       0.0
## fit2 -26.7       9.8
```

```r
bayes_R2(fit2)
```

```
##     Estimate  Est.Error      Q2.5     Q97.5
## R2 0.6162145 0.03108861 0.5475055 0.6672534
```

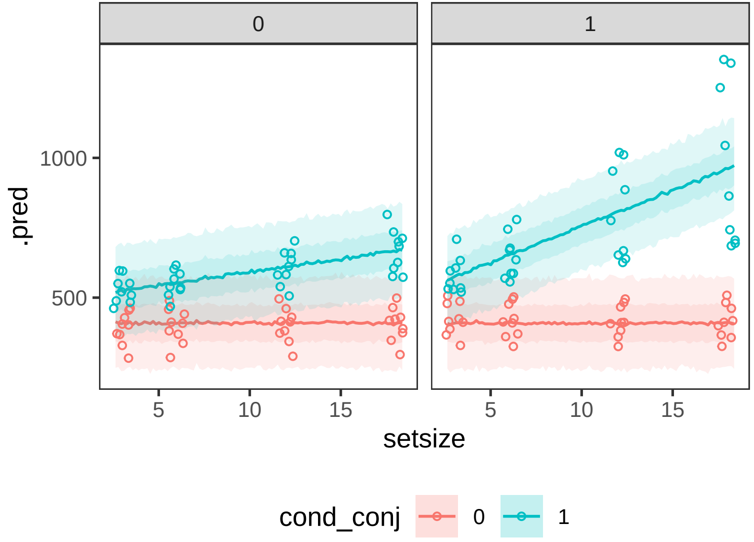
```r
bayes_R2(fit3)
```

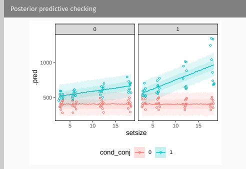
```
##     Estimate  Est.Error      Q2.5     Q97.5
## R2 0.7372387 0.01996627 0.6902033 0.7694252
```


Comparing the models
comp3 <- loo(fit3, fit2)
comp3$diffs
elpd_diff se_diff
fit3 0.0 0.0
fit2 -26.7 9.8
bayes_R2(fit2)
Estimate Est.Error Q2.5 Q97.5
R2 0.6162145 0.03108861 0.5475055 0.6672534
bayes_R2(fit3)
Estimate Est.Error Q2.5 Q97.5
R2 0.7372387 0.01996627 0.6902033 0.7694252

- this new model is better substantially better

Posterior predictive checking

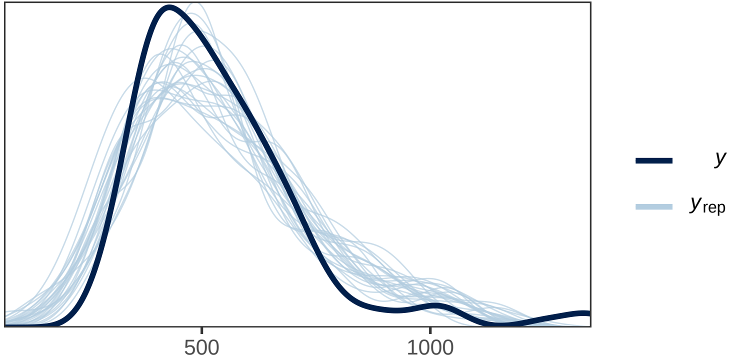




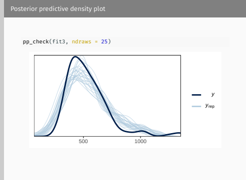
- But do see some misspecification (maybe see sign that sigma is non-constant)
- but also worse than that b/c points aren't even centered on regression line

Posterior predictive density plot

```r
pp_check(fit3, ndraws = 25)
```



2021-09-14



Can see the mis-specification here too, model predicts more low values below 250ms, fewer between 750 and 1000, and misses the bump at 1500

Inspecting the model fit

```r
summary(fit3)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: rt ~ setsize + cond_conj + setsize:cond_conj + setsize:cond_conj:targ_absent
##    Data: df2 (Number of observations: 144)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                               Estimate Est.Error l-95% CI u-95% CI Rhat
## Intercept                       406.52     23.03   361.78   450.21 1.00
## setsize                           0.17      2.01    -3.69     4.13 1.00
## cond_conj                        89.83     32.51    26.45   152.12 1.00
## setsize:cond_conj                 9.30      3.03     3.35    15.20 1.00
## setsize:cond_conj:targ_absent    16.30      2.03    12.33    20.36 1.00
##                               Bulk_ESS Tail_ESS
## Intercept                         2845     3017
## setsize                           2795     2837
## cond_conj                         2514     2605
## setsize:cond_conj                 2272     2489
## setsize:cond_conj:targ_absent     3575     3097
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    98.00      6.05    86.94   110.62 1.00     3292     2542
##
## Draws were sampled 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).
```


- should still remember that our parameters are conditional on the model.
- But now we are feeling pretty confident in the model, with the exception of σ, which shouldn't affect the β parameters too much
- can see setsize is about zero,
- setsize:cond_conj is 9, so when target is present each extra item adds average of 9ms, with some uncertainty
- three way interaction is 16, so when each extra item adds average of 9+16ms or 25ms, with some uncertainty

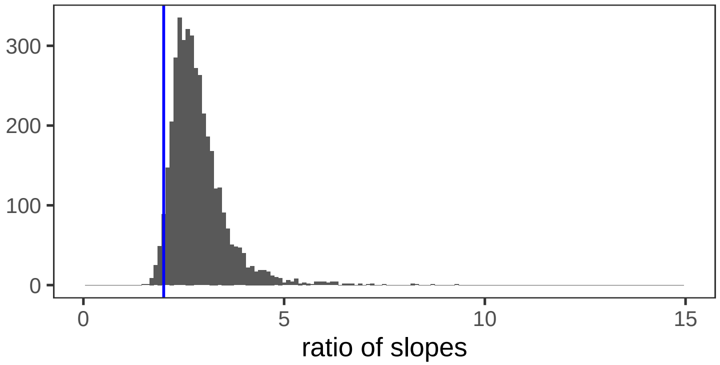
How does the influence of setsize differ for target present v. absent trials?

$$\frac{\text{RT per item (targ. absent)}}{\text{RT per item (targ. present)}} = \frac{\beta 2 + \beta_3 + \beta_4}{\beta_2 + \beta_3}$$

If this is a perfectly self-terminating search, we should expect this to be equal $\frac{n}{(n/2)} = 2$). Is it?

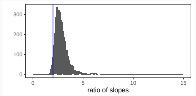
```
post_samps3 <- as_draws_df(fit3) %>%
  mutate(
    targ_absent_slope_ratio =
      (b_setsize + `b_setsize:cond_conj` + `b_setsize:cond_conj:targ_absent`) /
      (b_setsize + `b_setsize:cond_conj`)
    )
```

Visualizing the ratio of slopes



2021-09-14

Visualizing the ratio of slopes



Summarizing the ratio of slopes

```
Mode(post_samps3$targ_absent_slope_ratio)
```

```
## [1] 2.46689
```

```
rethinking::HPDI(post_samps3$targ_absent_slope_ratio, prob = .95)
```

```
##     |0.95    0.95|
## 1.798086 4.181386
```

```
mean(post_samps3$targ_absent_slope_ratio > 2)
```

```
## [1] 0.96775
```


Summarizing the ratio of slopes
Mode(post_samps3$targ_absent_slope_ratio)
[1] 2.46689
rethinking::HPDI(post_samps3$targ_absent_slope_ratio, prob = .95)
|0.95   0.95|
1.798086 4.181386
mean(post_samps3$targ_absent_slope_ratio > 2)
[1] 0.96775

Interpreting this finding

- It's unlikely people could be reliably identifying the target by searching fewer than $n/2$ items in the target present conditions
- More likely they are searching greater than n items in the target-absent condition, sometimes searching the same items multiple times.

Interpreting this finding
- It's unlikely people could be reliably identifying the target by searching fewer than $n/2$ items in the target present conditions
- More likely they are searching greater than n items in the target-absent condition, sometimes searching the same items multiple times.

Could this help explain the greater variability in rt for the target absent conjunction searches?

Maybe there are some individual differences we should look into later.

Another model: ANOVA

Compare with 2x2x4 ANOVA – these are different models

```
fit3_cat <- brm(
  rt ~ factor(setsize)*cond_conj*targ_absent,
  prior = prior(normal(500, 120), class="Intercept") +
    prior(normal(0, 360), class="b") +
    prior(exponential(.005), class="sigma"),
  data = df2
)
```

```
##      Estimate  Est.Error      Q2.5     Q97.5
## R2 0.7372387 0.01996627 0.6902033 0.7694252
```

```
##      Estimate  Est.Error      Q2.5     Q97.5
## R2 0.7286433 0.02111589 0.6818361 0.7625226
```

```
##          elpd_diff se_diff
## fit3      0.0       0.0
## fit3_cat -9.4       0.9
```


2021-09-14

Extra slides

Extra slides

Widely Applicable Information Criteria (WAIC)

- *WAIC* is a Bayesian generalization of *AIC* that also approximates leave-one-out cross validation.
- We estimate $\widehat{\text{elppd}}$ from the $\widehat{\text{lppd}}$ and the "effective number of parameters" $\widehat{p}_{\text{WAIC}}$.

$$\widehat{\text{elppd}}_{\text{WAIC}} = \widehat{\text{lppd}} - \widehat{p}_{\text{WAIC}}$$
$$\text{WAIC} = -2 \cdot \widehat{\text{elppd}}_{\text{WAIC}}$$

Widely Applicable Information Criteria (WAIC)
- *WAIC* is a Bayesian generalization of *AIC* that also approximates leave-one-out cross validation.
- We estimate $\widehat{\text{elppd}}$ from the $\widehat{\text{lppd}}$ and the "effective number of parameters" $\widehat{p}_{\text{WAIC}}$.
$\widehat{\text{elppd}}_{\text{WAIC}} = \widehat{\text{lppd}} - \widehat{p}_{\text{WAIC}}$
$\text{WAIC} = -2 \cdot \widehat{\text{elppd}}_{\text{WAIC}}$