Amazon Reviews

Dr. Jarad Niemi

STAT 544 - Iowa State University

March 5, 2018

Amazon Reviews - Upright, bagless, cyclonic vacuum cleaners

	Number of ratings							
product_id	n1	n2	n3	n4	n5	n_total	mean	sd
B000REMVGK	21	17	2	8	7	55	2.33	1.44
B001EFMD8W	40	34	28	77	347	526	4.25	1.26
B001PB51GQ	14	12	13	31	69	139	3.93	1.36
B002DGSJVG	22	8	3	6	10	49	2.47	1.63
B002G9UQZC	8	0	1	1	1	11	1.82	1.47
B002GHBRX4	18	8	9	14	27	76	3.32	1.61
B002HF66BI	9	5	2	2	3	21	2.29	1.49
B003OA77MC	15	7	8	24	42	96	3.74	1.47
B003OAD24Y	7	7	4	9	19	46	3.57	1.53
B003Y3AA3C	20	3	1	2	2	28	1.68	1.28
B0043EW354	40	25	25	60	163	313	3.90	1.44
B00440EO8G	2	1	1	1	7	12	3.83	1.64
B004R9197I	9	1	1	9	26	46	3.91	1.58
B008L5F4H0	3	1	2	12	7	25	3.76	1.27

Model for Amazon Reviews

Let y_{pr} be the rth review for the pth product. Assume

$$y_{pr} \stackrel{ind}{\sim} N(\theta_p, \sigma^2)$$

and

$$\theta_p \stackrel{ind}{\sim} N(\mu, \tau^2)$$

and

$$p(\mu, \tau, \sigma) \propto Ca^+(\sigma; 0, 1) Ca^+(\tau; 0, 1)$$

Model parameterization convenient for Stan/JAGS

Let

- Y_i be number of stars for review i and
- $p[i]$ be the numeric product id for review i.

Then the model can be rewritten as

$$Y_i \stackrel{ind}{\sim} N(\theta_{p[i]}, \sigma^2)$$

and the hierarchical portion is

$$\theta_p \stackrel{ind}{\sim} N(\mu, \tau^2)$$

and the prior is

$$p(\mu, \tau, \sigma) \propto Ca^+(\sigma; 0, 1) Ca^+(\tau; 0, 1).$$

Normal hierarchical model in Stan

```
normal_model = "
data {
  int <lower=1> n;
  int <lower=1> n_products;
  int <lower=1,upper=5> stars[n];
  int <lower=1,upper=n_products> product_id[n];
}

parameters {
  real mu;                 // implied uniform prior
  real<lower=0> sigma;
  real<lower=0> tau;
  real theta[n_products];
}

model {
  // Prior
  sigma ~ cauchy(0,1);
  tau   ~ cauchy(0,1);

  // Hierarchial model
  theta ~ normal(mu,tau);

  // Data model
  for (i in 1:n) stars[i] ~ normal(theta[product_id[i]], sigma);
}
"
```

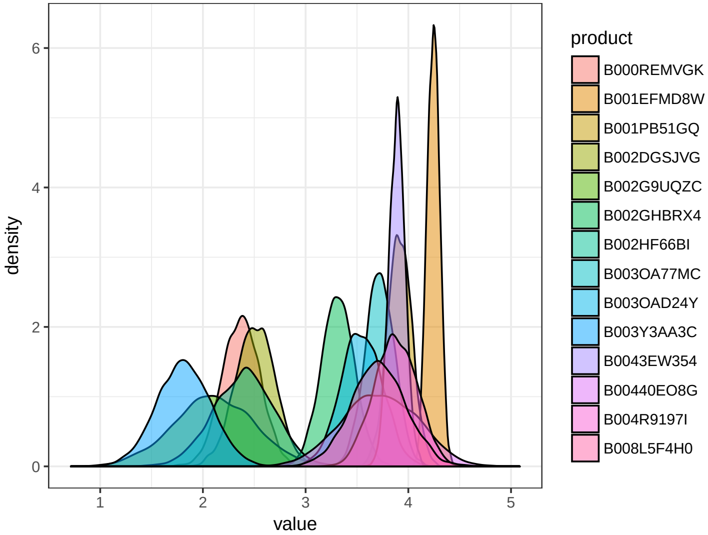
Fit model

```
m = stan_model(model_code = normal_model)


In file included from file59626513b0bb.cpp:8:
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/src/st
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/math/tool
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config.hp
/Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config/compiler/clang.hpp:200:1
#  define BOOST_NO_CXX11_RVALUE_REFERENCES
          ^
<command line>:6:9: note: previous definition is here
#define BOOST_NO_CXX11_RVALUE_REFERENCES 1
        ^
1 warning generated.


dat = list(n = nrow(d),
           n_products = nlevels(d$product_id),
           stars = d$stars,
           product_id = as.numeric(d$product_id))
r = sampling(m, dat)



SAMPLING FOR MODEL '03148bf3617900613206f68b66119d86' NOW (CHAIN 1).

Gradient evaluation took 0.000276 seconds
```

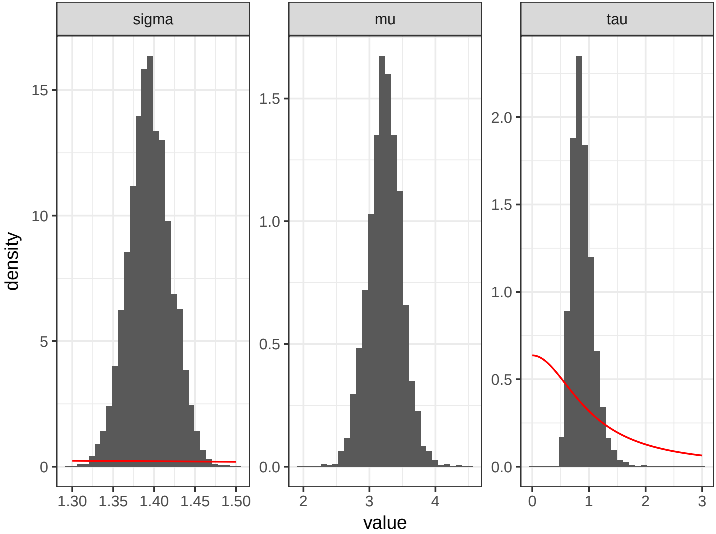
Tabular summary

```
Inference for Stan model: 03148bf3617900613206f68b66119d86.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

              mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff Rhat
mu            3.23    0.00 0.26     2.73     3.07     3.23     3.40     3.73  4000    1
sigma         1.39    0.00 0.03     1.34     1.38     1.39     1.41     1.45  4000    1
tau           0.89    0.00 0.19     0.58     0.75     0.86     0.99     1.34  4000    1
theta[1]      2.37    0.00 0.18     2.02     2.25     2.37     2.49     2.72  4000    1
theta[2]      4.24    0.00 0.06     4.13     4.20     4.25     4.29     4.36  4000    1
theta[3]      3.92    0.00 0.12     3.68     3.84     3.91     3.99     4.15  4000    1
theta[4]      2.51    0.00 0.19     2.14     2.38     2.51     2.64     2.88  4000    1
theta[5]      2.10    0.01 0.39     1.33     1.84     2.10     2.37     2.86  4000    1
theta[6]      3.31    0.00 0.16     3.00     3.21     3.31     3.42     3.63  4000    1
theta[7]      2.40    0.00 0.29     1.82     2.20     2.40     2.59     2.95  4000    1
theta[8]      3.72    0.00 0.14     3.45     3.63     3.72     3.82     4.00  4000    1
theta[9]      3.54    0.00 0.20     3.15     3.41     3.54     3.68     3.93  4000    1
theta[10]     1.81    0.00 0.26     1.30     1.63     1.81     1.99     2.33  4000    1
theta[11]     3.89    0.00 0.08     3.74     3.84     3.89     3.94     4.05  4000    1
theta[12]     3.72    0.01 0.36     3.01     3.47     3.72     3.98     4.42  4000    1
theta[13]     3.88    0.00 0.21     3.47     3.73     3.87     4.02     4.28  4000    1
theta[14]     3.71    0.00 0.27     3.19     3.53     3.71     3.89     4.23  4000    1
lp__      -1207.37    0.07 2.87 -1213.62 -1209.10 -1207.11 -1205.33 -1202.55  1515    1

Samples were drawn using NUTS(diag_e) at Mon Mar  5 16:42:40 2018.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Vacuum cleaner mean posteriors (θ_p)



Other parameter posteriors



A quick rating

Suppose a new vacuum cleaner comes on the market and there are two Amazon reviews both with 5 stars. What do you think the average star rating will be (in the future) for this new product?

Let n^* be the number of new ratings and $\overline{y}^*$ be the average of those ratings, then

$$\begin{aligned} E[\theta^*|\overline{y}^*, n^*, \sigma, \mu, \tau] &= \frac{\frac{n^*}{\sigma^2}}{\frac{n^*}{\sigma^2}+\frac{1}{\tau^2}}\overline{y}^* + \frac{\frac{1}{\tau^2}}{\frac{n^*}{\sigma^2}+\frac{1}{\tau^2}}\mu \\ &= \frac{n^*}{n^*+\frac{\sigma^2}{\tau^2}}\overline{y}^* + \frac{\frac{\sigma^2}{\tau^2}}{n^*+\frac{\sigma^2}{\tau^2}}\mu \\ &= \frac{n^*}{n^*+m}\overline{y}^* + \frac{m}{n^*+m}\mu \end{aligned}$$

where $m = \sigma^2/\tau^2$ is a measure of how many *prior* samples there are.

IMDB rating

From `http://www.imdb.com/chart/top.html`:

```
weighted rating (WR) = (v / (v+m))  R + (m / (v+m))  C

Where:

R = average for the movie (mean) = (Rating)
v = number of votes for the movie = (votes)
m = minimum votes required to be listed in the Top 250
    (currently 25000)
C = the mean vote across the whole report (currently 7.1)
```

Thus IMDB uses a Bayesian estimate for the rating for each movie where $m = \sigma^2/\tau^2 = 25,000$. IMDB has enough data that the uncertainty in $\mu(C)$, σ^2, and τ^2 is pretty minimal.

Clearly incorrect model

We assumed

$$y_{rp} \stackrel{ind}{\sim} N(\theta_p, \sigma^2)$$

for the rth star rating of product p. Clearly this model is incorrect since $y_{ij} \in \{1, 2, 3, 4, 5\}$.

An alternative model is

$$z_{ij} \stackrel{ind}{\sim} Bin(4, \theta_p)$$

where $z_{ij} = y_{ij} - 1$ is the jth star rating minus 1 of product i and

$$\theta_p \sim Be(\alpha, \beta) \qquad \text{and} \qquad p(\alpha, \beta) \propto (\alpha + \beta)^{-5/2}.$$

The idea behind this model would be that product i the probability of earning each star is θ_p and each star is independent.

Binomial hierarchical model in Stan

```
binomial_model = "
data {
  int <lower=1> n;
  int <lower=1> n_products;
  int <lower=1,upper=5> stars[n];
  int <lower=1,upper=n_products> product_id[n];
}

transformed data {
  int <lower=0, upper=4> z[n];
  for (i in 1:n) z[i] = stars[i]-1;
}

parameters {
  real<lower=0> alpha;
  real<lower=0> beta;
  real<lower=0,upper=1> theta[n_products];
}

model {
  // Prior
  target += -5*log(alpha+beta)/2; // improper prior

  // Hierarchical model
  theta ~ beta(alpha,beta);

  // Data model
  for (i in 1:n) z[i] ~ binomial(4, theta[product_id[i]]);
}
"
```

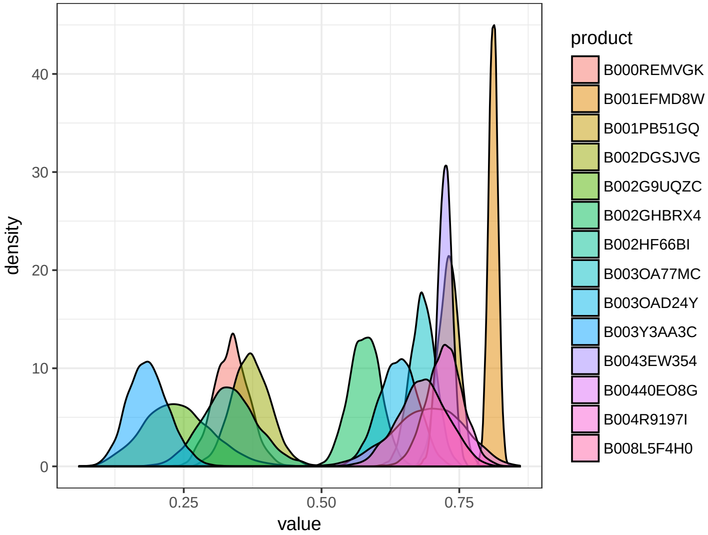
Fit model

```
m = stan_model(model_code = binomial_model)


In file included from file596211f491db.cpp:8:
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/src/st
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/math/tool
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config.hp
/Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config/compiler/clang.hpp:200:1
#  define BOOST_NO_CXX11_RVALUE_REFERENCES
          ^
<command line>:6:9: note: previous definition is here
#define BOOST_NO_CXX11_RVALUE_REFERENCES 1
        ^
1 warning generated.


dat = list(n = nrow(d),
           n_products = nlevels(d$product_id),
           stars = d$stars,
           product_id = as.numeric(d$product_id))
r = sampling(m, dat)



SAMPLING FOR MODEL 'e26b5a276955604814aba1dc21dc3cbe' NOW (CHAIN 1).

Gradient evaluation took 0.000358 seconds
```

Tabular summary

```
Inference for Stan model: e26b5a276955604814aba1dc21dc3cbe.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

              mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff Rhat
alpha         2.71    0.02 1.09     1.05     1.92     2.56     3.33     5.21  3617    1
beta          2.28    0.01 0.87     0.94     1.64     2.15     2.78     4.29  3744    1
theta[1]      0.34    0.00 0.03     0.27     0.31     0.34     0.36     0.40  4000    1
theta[2]      0.81    0.00 0.01     0.79     0.81     0.81     0.82     0.83  4000    1
theta[3]      0.73    0.00 0.02     0.69     0.72     0.73     0.74     0.77  4000    1
theta[4]      0.37    0.00 0.03     0.30     0.35     0.37     0.39     0.44  4000    1
theta[5]      0.24    0.00 0.06     0.13     0.20     0.24     0.28     0.37  4000    1
theta[6]      0.58    0.00 0.03     0.52     0.56     0.58     0.60     0.63  4000    1
theta[7]      0.33    0.00 0.05     0.24     0.30     0.33     0.37     0.44  4000    1
theta[8]      0.68    0.00 0.02     0.64     0.67     0.68     0.70     0.73  4000    1
theta[9]      0.64    0.00 0.03     0.57     0.62     0.64     0.66     0.70  4000    1
theta[10]     0.19    0.00 0.04     0.12     0.16     0.18     0.21     0.26  4000    1
theta[11]     0.72    0.00 0.01     0.70     0.72     0.72     0.73     0.75  4000    1
theta[12]     0.69    0.00 0.06     0.56     0.65     0.70     0.74     0.81  4000    1
theta[13]     0.72    0.00 0.03     0.66     0.70     0.72     0.75     0.79  4000    1
theta[14]     0.68    0.00 0.05     0.59     0.65     0.68     0.71     0.77  4000    1
lp__      -3265.27    0.07 2.85 -3271.73 -3266.90 -3264.94 -3263.23 -3260.57  1489    1

Samples were drawn using NUTS(diag_e) at Mon Mar  5 16:44:25 2018.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Review mean posteriors (θ_p)



Other parameter posteriors

Recall that

- α is the prior success
- β is the prior failures

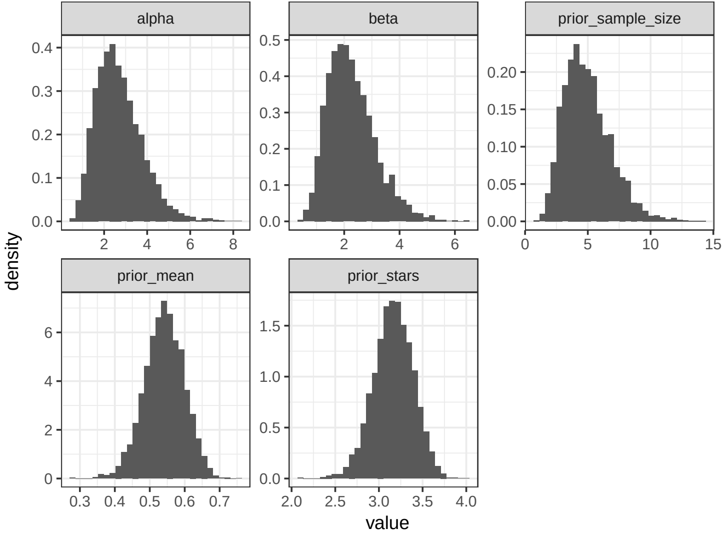
So

- $\alpha + \beta$ is the prior sample size
- $E[\theta_p|\alpha, \beta] = \frac{\alpha}{\alpha+\beta}$ is the prior expectation for the probability

But we might want to show results on the original scale (stars), so the expected number of stars for a new product is

$$\begin{aligned} E[\text{stars}_{*j}|\alpha, \beta] &= E[z_{*j} + 1|\alpha, \beta] = E[z_{*j}|\alpha, \beta] + 1 \\ &= E[E[z_{*j}|\theta^*]|\alpha, \beta] + 1 = E[4\theta^*|\alpha, \beta] + 1 \\ &= 4\frac{\alpha}{\alpha+\beta} + 1 \end{aligned}$$

Other parameter posteriors



Uniform use of star ratings

This binomial model has the proper support $\{0, 1, 2, 3, 4\}$ for stars minus 1, but does it have the correct proportion of observations in each star category?

As an example, $\hat{\theta}_2 = 0.81$. Thus, we would expect if we used $\hat{\theta}_2$

stars	theoretical	observed
1	0.001	0.076
2	0.022	0.065
3	0.142	0.053
4	0.404	0.146
5	0.430	0.660

But this ignores the uncertainty in θ_2 (95% CI is (0.79, 0.83)), so perhaps this difference is due to this uncertainty.

Posterior predictive pvalue

To assess this model fit, we will simulate posterior predictive star ratings for product 2 and compare to the observed ratings:

product_id	n1	n2	n3	n4	n5	n_total
B001EFMD8W	40	34	28	77	347	526

Let $\tilde{z}_2$ be all the predictive data for product 2, i.e. $\tilde{z}_2 = (\tilde{z}_{21}, \ldots, \tilde{z}_{2J})$ with $J = 526$ where $\tilde{z}_{2j}$ is the jth predictive star rating minus 1 for review j of product 2. Then

$$p(\tilde{z}_2|z) = \int \left[\prod_{j=1}^{J} p(\tilde{z}_{2j}|\theta_2) \right] p(\theta_2|z) d\theta_2$$

Thus the following procedure will simulation from the joint distribution for the predictive ratings:

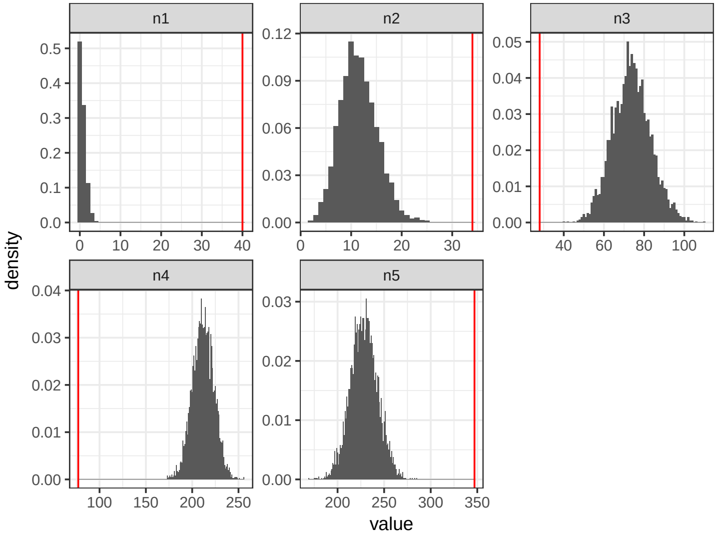
1. $\theta_2 \sim p(\theta_2|z)$,
2. For $j = 1, \ldots, 526$, $z_{2j} \stackrel{ind}{\sim} Bin(4, \theta_2)$, and
3. star$_{2j} = z_{2j} + 1$.

Posterior predictive distribution in R

```
theta2 = as.numeric(draws$theta[,2])

ztilde2 = plyr::adply(theta2, 1, function(x) {
  ztilde = rbinom(526, 4, x) + 1
  data.frame(n1 = sum(ztilde==1),
             n2 = sum(ztilde==2),
             n3 = sum(ztilde==3),
             n4 = sum(ztilde==4),
             n5 = sum(ztilde==5))
})
head(ztilde2)
```

```
  X1 n1 n2 n3  n4  n5
1  1  1 16 77 182 250
2  2  0 10 83 213 220
3  3  0  8 76 231 211
4  4  0 11 77 225 213
5  5  0 20 96 210 200
6  6  0  9 70 221 226
```

Posterior predictive distribution in R



Ordinal data model

Let $s_p = (s_{i1}, \ldots, s_{i5})$ be the vector of the number of 1-star to 5-star ratings for product i, assume

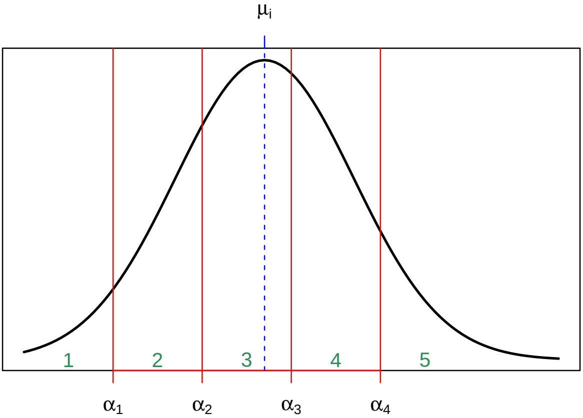
$$S_i \stackrel{ind}{\sim} Mult(n_p, \theta_p)$$

where θ_p is a probability vector

$$\theta_{ik} = \int_{\alpha_{k-1}}^{\alpha_k} N(x|\mu_p, 1)dx = \Phi(\alpha_k - \mu_p) - \Phi(\alpha_{k-1} - \mu_p)$$

where $\alpha_0 = -\infty$, $\alpha_1 = 0$, and $\alpha_5 = \infty$, and Φ is the standard normal cumulative distribution function (cdf).

Visualizing the model



Hierarchical model

So each product has its own mean μ_p. The larger μ_p is the more 5-star ratings the product will receive and the fewer 1-star ratings the product will review.

In order to borrow information across different products, we might assume a hierarchical model for the μ_p, e.g.

$$\mu_p \stackrel{ind}{\sim} N(\eta, \tau^2)$$

with a prior

$$p(\eta, \tau) \propto Ca(\tau; 0, 1).$$

```
ordinal_model = "
data {
  int <lower=1> n_products;
  int <lower=0> s[n_products,5]; // summarized count by product
}

parameters {
  real<lower=0> alpha_diff[3];
  real mu[n_products];
  real eta;
  real<lower=0> tau;
}

transformed parameters {
  ordered[4] alpha;              // cut points
  simplex[5] theta[n_products]; // each theta vector sums to 1

  alpha[1] = 0; for (i in 1:3) alpha[i+1] = alpha[i] + alpha_diff[i];

  for (p in 1:n_products) {
    theta[p,1] = Phi(-mu[p]);
    for (j in 2:4)
      theta[p,j] = Phi(alpha[j]-mu[p]) - Phi(alpha[j-1]-mu[p]);
    theta[p,5] = 1-Phi(alpha[4]-mu[p]);
  }
}

model {
  tau ~ cauchy(0,1);
  mu ~ normal(eta, tau);
  for (p in 1:n_products) s[p] ~ multinomial(theta[p]); // n_reviews[p] is implicit
}
"
```

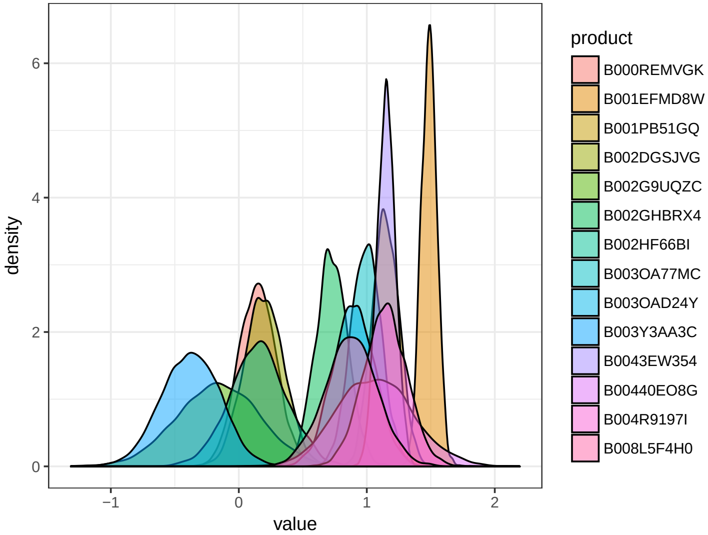
Fit model

```
m = stan_model(model_code = ordinal_model)


In file included from file59623973d09b.cpp:8:
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/src/st
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/math/tool
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config.hp
/Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/config/compiler/clang.hpp:200:1
#  define BOOST_NO_CXX11_RVALUE_REFERENCES
          ^
<command line>:6:9: note: previous definition is here
#define BOOST_NO_CXX11_RVALUE_REFERENCES 1
        ^
1 warning generated.


dat = list(n_products = nrow(for_table),
           s = as.matrix(for_table[,2:6]))
r = sampling(m, dat, pars = c("alpha","eta","tau","mu"))



SAMPLING FOR MODEL 'cfd399bb3e758fc22eaf105a07c2068f' NOW (CHAIN 1).

Gradient evaluation took 9.2e-05 seconds
1000 transitions using 10 leapfrog steps per transition would take 0.92 seconds.
Adjust your expectations accordingly!
```

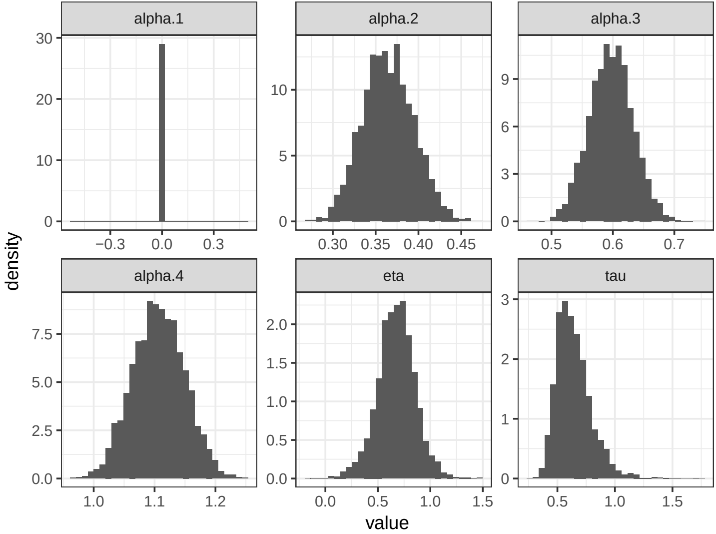
Fit model

```
r


Inference for Stan model: cfd399bb3e758fc22eaf105a07c2068f.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

             mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff Rhat
alpha[1]     0.00    0.00 0.00     0.00     0.00     0.00     0.00     0.00  4000  NaN
alpha[2]     0.36    0.00 0.03     0.31     0.34     0.36     0.38     0.42  4000    1
alpha[3]     0.60    0.00 0.04     0.53     0.57     0.60     0.62     0.67  3484    1
alpha[4]     1.11    0.00 0.04     1.02     1.08     1.11     1.14     1.19  3191    1
eta          0.68    0.00 0.18     0.30     0.56     0.68     0.79     1.03  4000    1
tau          0.64    0.00 0.15     0.42     0.53     0.62     0.72     0.99  3554    1
mu[1]        0.15    0.00 0.14    -0.13     0.05     0.15     0.24     0.43  4000    1
mu[2]        1.49    0.00 0.06     1.37     1.44     1.49     1.53     1.61  4000    1
mu[3]        1.15    0.00 0.10     0.95     1.08     1.15     1.22     1.35  4000    1
mu[4]        0.20    0.00 0.15    -0.10     0.09     0.20     0.30     0.49  4000    1
mu[5]       -0.16    0.01 0.32    -0.79    -0.38    -0.16     0.06     0.44  4000    1
mu[6]        0.73    0.00 0.13     0.48     0.64     0.72     0.81     0.98  4000    1
mu[7]        0.15    0.00 0.22    -0.29     0.01     0.15     0.30     0.59  4000    1
mu[8]        0.99    0.00 0.12     0.76     0.91     1.00     1.07     1.23  4000    1
mu[9]        0.90    0.00 0.16     0.58     0.79     0.90     1.01     1.22  4000    1
mu[10]      -0.38    0.00 0.23    -0.83    -0.53    -0.37    -0.22     0.06  4000    1
mu[11]       1.15    0.00 0.07     1.01     1.10     1.15     1.20     1.29  4000    1
mu[12]       1.06    0.00 0.29     0.52     0.86     1.06     1.26     1.66  4000    1
mu[13]       1.14    0.00 0.17     0.81     1.03     1.14     1.26     1.47  4000    1
mu[14]       0.88    0.00 0.20     0.47     0.74     0.88     1.01     1.28  4000    1
lp__     -1835.61    0.07 3.03 -1842.26 -1837.41 -1835.37 -1833.46 -1830.40  2011    1

Samples were drawn using NUTS(diag_e) at Mon Mar  5 16:45:54 2018.
```

Review mean posteriors (θ_p)



Other parameter posteriors



Visualizing the model