Linear Models with the rstanarm R Package

Ben Goodrich

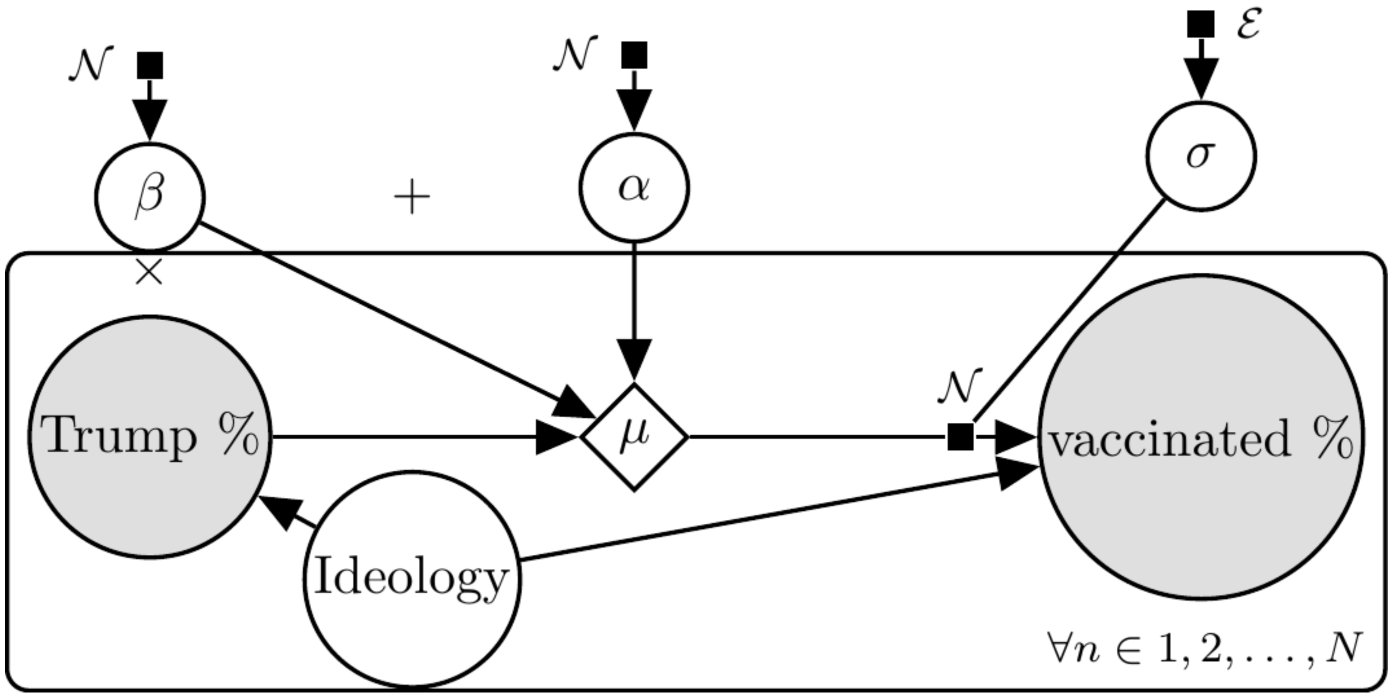
March 21, 2022

Data on 2020 Trump Vote and 2022 Vaccination

```
library(readr); library(dplyr)
# https://docs.google.com/spreadsheets/d/100BFc0VppVL8CIhaNh5ZiTFGBNCnGBdYzfqISAWxln8/
Gabba <- read_csv("Gabba.csv", col_types = c("ccccdddddddddd"), skip = 1, col_names =
                    c("FIPS", "ST", "State", "County", "Trump#", "Votes#", "Trump", "Pop",
                      "Vaccinated#", "Vaccinated", "Death1", "Death2", "Death3", "Death4"))
select(Gabba, State:Vaccinated) %>%
  glimpse # each row is a county
```

```
## Rows: 3,144
## Columns: 8
## $ State         <chr> "Alabama", "Alabama", "Alabama", "Alabama", "Alabama", "Alabama", …
## $ County        <chr> "Autauga", "Baldwin", "Barbour", "Bibb", "Blount", "Bullock", "But…
## $ `Trump#`      <dbl> 19838, 83544, 5622, 7525, 24711, 1146, 5458, 35101, 8753, 10583, 1…
## $ `Votes#`      <dbl> 27770, 109679, 10518, 9595, 27588, 4613, 9488, 50983, 15284, 12301…
## $ Trump         <dbl> 71.44, 76.17, 53.45, 78.43, 89.57, 24.84, 57.53, 68.85, 57.27, 86.…
## $ Pop           <dbl> 58805, 231767, 25223, 22293, 59134, 10357, 19051, 116441, 34772, 2…
## $ `Vaccinated#` <dbl> 24395, 112300, 11070, 7728, 18162, 5305, 7613, 52780, 10354, 7964,…
## $ Vaccinated    <dbl> 41.48, 48.45, 43.89, 34.67, 30.71, 51.22, 39.96, 45.33, 29.78, 31.…
```

Model for Vaccinated % by U.S. County



Model

- Circles are variables, shading indicates the variable is observable, diamond indicates the quantity is deterministic, plates indicate that all of the interior quantities are indexed by n, squares indicate probability distributions

Prior Predictive Distribution for a Linear Model

$$\alpha \sim ???$$

$$\forall k : \beta_k \sim ???$$

$$\forall n : \mu_n \equiv \alpha + \sum_{k=1}^{K} \beta_k x_{nk}$$

$$0 < \sigma \sim ???$$

$$\forall n : \epsilon_n \sim \mathcal{N}\left(0, \sigma\right)$$

$$\forall n : y_n \equiv \mu_n + \epsilon_n$$

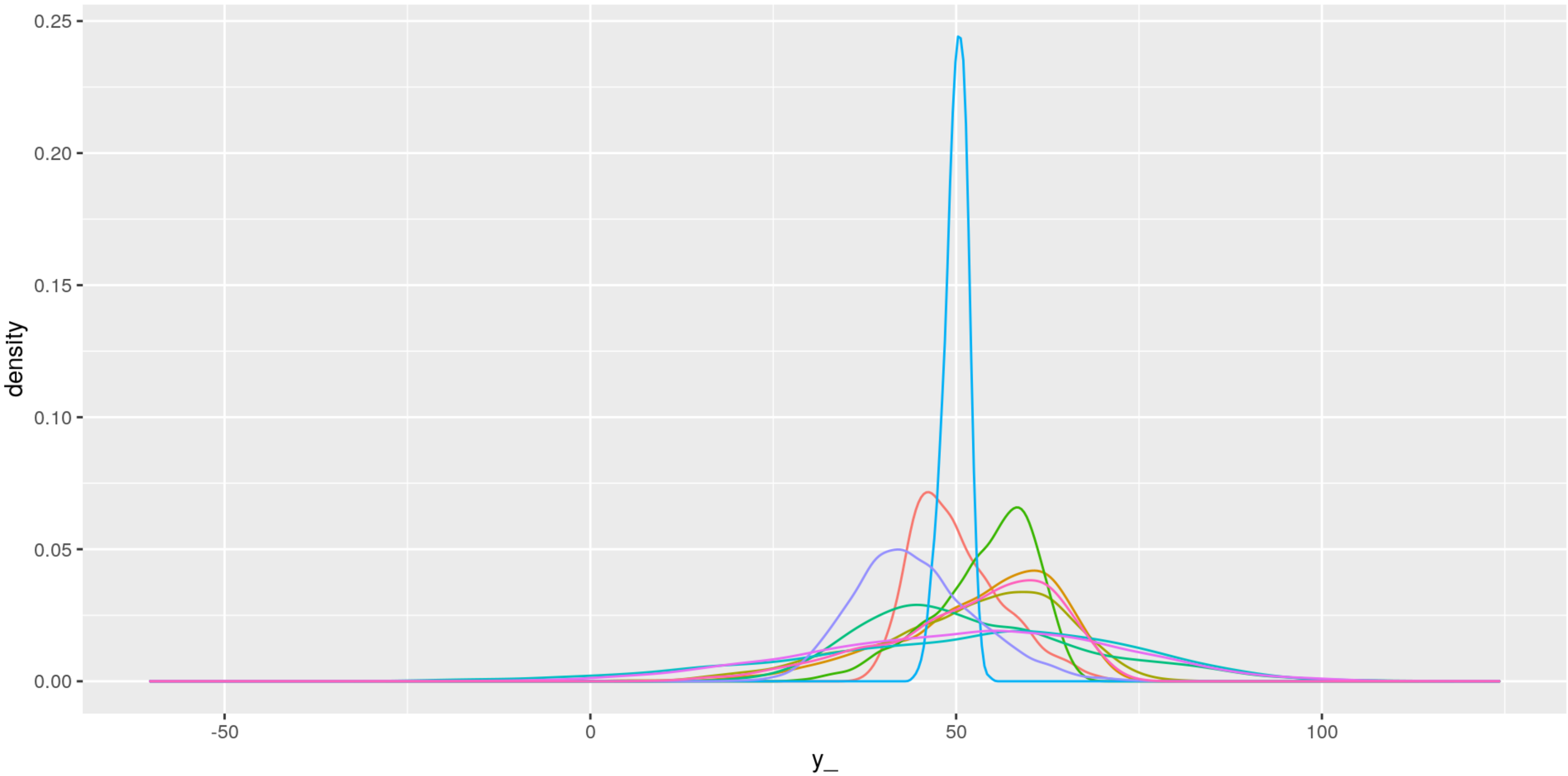
where `???` indicates the parameter is drawn from your belief distribution

- The assumption of this data-generating process is that each ϵ_n is MARGINALLY normal (with expectation 0 and standard deviation $\sigma > 0$), implying that each y_n is CONDITIONALLY normal (with expectation μ_n and standard deviation σ)
- You can mimic what happens when you put a probability distribution through an assumed data-generating process by putting a large number of random draws from that probability distribution through that data-generating process

Drawing from the Prior Predictive Distribution

```r
N <- nrow(Gabba); x <- Gabba$Trump; x <- x - mean(x, na.rm = TRUE)
prior_predictive <- replicate(10, { # usually do more draws, but harder to plot
  alpha_ <- rnorm(n = 1, mean = 50, sd = 3)  # relative to a centered x
  beta_ <- rnorm(n = 1,  mean = -0.5, sd = 1)
  mu_ <- alpha_ + beta_ * x

  sigma_ <- rexp(n = 1, rate = 0.2)
  epsilon_ <- rnorm(n = N, mean = 0, sd = sigma_)

  y_ <- mu_ + epsilon_
  return(y_)
})

library(ggplot2) # plot on next slide
ggplot(tibble(y_ = c(prior_predictive), replication = as.factor(rep(1:10, each = N)))) +
  geom_density(aes(x = y_, color = replication), show.legend = FALSE)
```

Plot from Previous Slide (each line is a dataset)



The `stan_glm` Function in the rstanarm Package

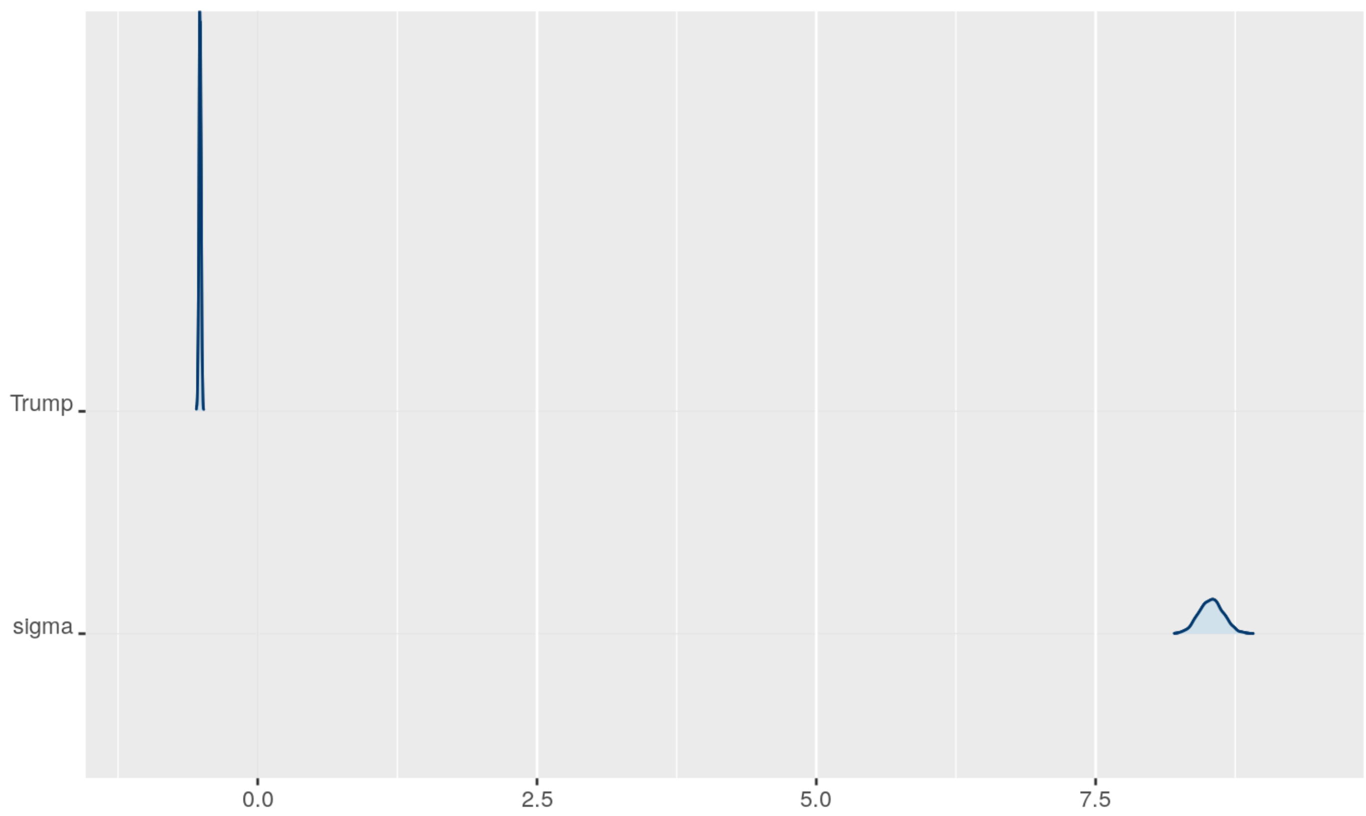
```
post <- stan_glm(Vaccinated ~ Trump, data = Gabba, subset = Vaccinated <= 100,
                 family = gaussian, cores = 4, seed = 12345, # set.seed() insufficient
                 prior_intercept = normal(location = 50, scale = 3),
                 prior = normal(location = -0.5, scale = 1),
                 prior_aux = exponential(rate = 0.2)) # expectation and std. deviation of 5
```

```
post # intercept relative to uncentered predictors, i.e. a county where Trump got 0%
```

```
...
## ------
##             Median MAD_SD
## (Intercept) 84.6    0.6
## Trump       -0.5    0.0
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 8.5    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
...
```

Plotting the Marginal Posterior Densities

```
plot(post, plotfun = "areas_ridges", pars = c("Trump", "sigma")) # excluding intercept
```



Credible Intervals and Posterior Probabilities

```
# what people mistake confidence intervals for
round(posterior_interval(post, prob = 0.8), digits = 2)
```

```
##               10%   90%
## (Intercept) 83.82 85.38
## Trump       -0.53 -0.50
## sigma        8.40  8.68
```

```
beta <- as.data.frame(post)$Trump # coefficient
mean(beta > -0.5) # what people mistake p-values for
```

```
## [1] 0.04575
```

Do This Once on Each Computer You Use

- R comes with a terrible default coding for ordered factors in regressions known as “Helmert” contrasts
- Execute this once to change them to “treatment” contrasts, which is the conventional coding in the social sciences with dummy variables relative to a baseline category

```
cat('options(contrasts = c(unordered = "contr.treatment", ordered = "contr.treatment"))',
    file = "~/.Rprofile", sep = "\n", append = TRUE)
```

- Without this, you will get a weird rotation of the coefficients on dummy variables made from unordered factors
- `"contr.sum"` is another reasonable (but rare) choice

The `stan_lm` Function in the rstanarm Package

- Suppose you wanted to include a dummy variable for each state (but one)
- One option is to give the state shifts a normal distribution with expectation zero and unknown standard deviation, δ, which has its own prior
- That would be problematic because your beliefs about the shift in Alabama really should not be independent of your beliefs about the shift in Mississippi
- `stan_lm` instead asks you for a beta prior on the R^2, with first shape parameter $\frac{K+1}{2}$. Specifying a prior mode (the default), mean or median determines the second shape parameter, and the coefficients have a joint prior with maximum entropy given the $\mathbb{E}\left[\ln R^2\right]$.
- There is a Jeffreys' prior on the marginal standard deviation of the outcome, which along with the R^2 determines the standard deviation of the error

```
post <- stan_lm(Vaccinated ~ Trump + State, data = Gabba, subset = Vaccinated <= 100,
                prior = R2(0.75), prior_intercept = normal(location = 50, scale = 3))
```

Why NUTS Is Better than Other MCMC Samplers

- With Stan, it is almost always the case that things either go well or you get valid warning messages
- Because Stan uses gradients, it scales well as models get more complex. It tends to be the case that the first-order autocorrelation is negative so you can get greater effective sample sizes for means.

```
round(bayesplot::neff_ratio(post)[-(6:48)], digits = 2)
```

```
##        (Intercept)              Trump        StateAlaska       StateArizona
##               0.69               0.83               1.15               1.22
##      StateArkansas    StateWashington StateWest Virginia     StateWisconsin
##               0.95               1.02               1.18               1.20
##       StateWyoming              sigma      log-fit_ratio                 R2
##               1.30               1.69               1.10               0.86
```

Posterior Prediction

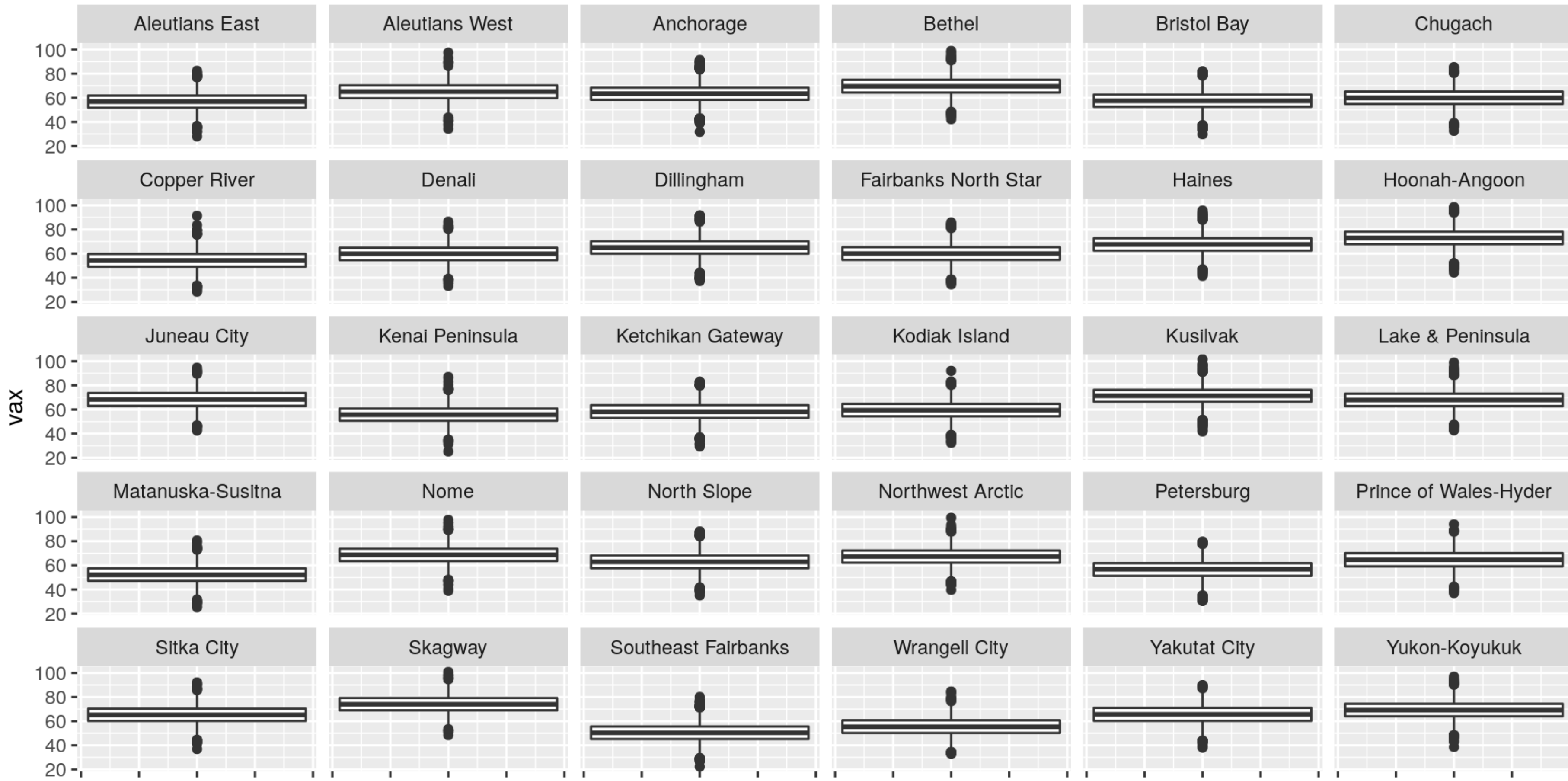
$$f\left(y_{n+1} \mid y_1, \dots y_n, x_{n+1}\right) = f\left(y_{n+1} \bigcap \cancel{\alpha} \bigcap \cancel{\boldsymbol{\beta}} \bigcap \cancel{\sigma} \mid y_1, \dots y_n, x_{n+1}\right) =$$

$$\int_0^\infty \int_{-\infty}^\infty \int_{-\infty}^\infty f\left(y_{n+1} \mid \alpha, \boldsymbol{\beta}, \sigma, x_{n+1}\right) f\left(\alpha, \boldsymbol{\beta}, \sigma \mid y_1, \dots, y_n\right) d\alpha d\boldsymbol{\beta} d\sigma$$

We typically cannot evaluate those definite integrals, but we can draw S times from the distribution whose PDF is $f\left(y_{n+1} \mid y_1, \dots y_n\right)$ by drawing S times from the posterior distribution of the parameters given the past data and using each of those realizations of the parameters to draw y_{n+1} from its conditional distribution:

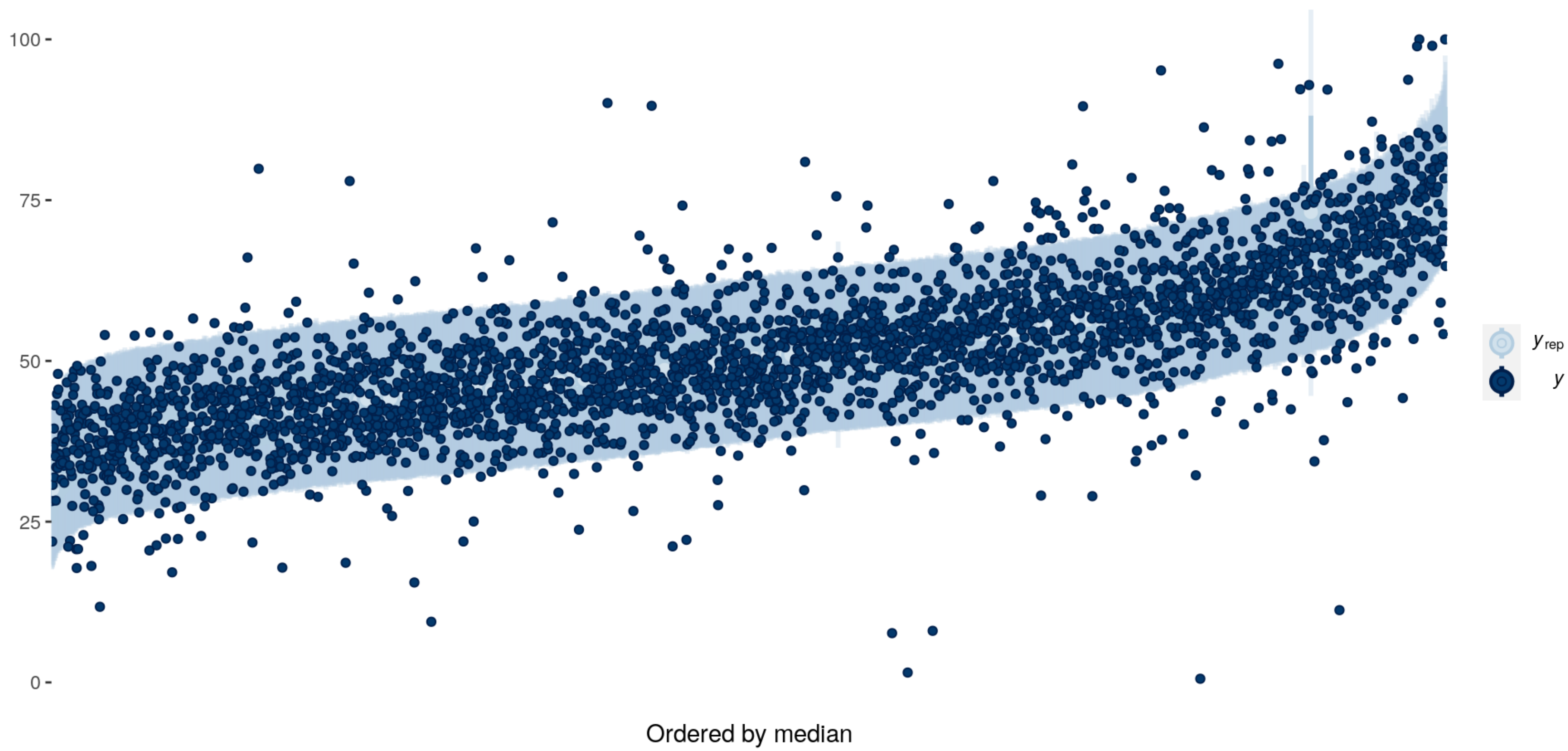
```
Alaska <- filter(Gabba, State == "Alaska") %>% na.omit
PPD <- posterior_predict(post, newdata = Alaska) # draws x counties (4000 x 30)
ggplot(tibble(vax = c(PPD), county = rep(Alaska$County, each = nrow(PPD)))) +
  geom_boxplot(aes(y = vax)) + facet_wrap(~county) + theme(axis.text.x = element_blank())
```

Plot from Previous Slide



Posterior Predictive Checking

```
pp_check(post, plotfun = "loo_intervals", order = "median") # each dot is a county
```



Excercise: IQ of Three Year Olds

- Many rstanarm examples are available at https://avehtari.github.io/ROS-Examples/examples.html
- At 36 months, kids were given an IQ test
- Suppose the conditional expectation is a linear function of variables pertaining to the mother

```
data(kidiq, package = "rstanarm")
colnames(kidiq)
```

```
## [1] "kid_score" "mom_hs"    "mom_iq"    "mom_age"
```

```
mom_hs  <- kidiq$mom_hs - mean(kidiq$mom_hs)
mom_iq  <- (kidiq$mom_iq - mean(kidiq$mom_iq)) / 10   # units are 10-points, not points
mom_age <- (kidiq$mom_age - mean(kidiq$mom_age)) / 10 # units are decades, not years
```

Drawing from the Prior Predictive Distribution

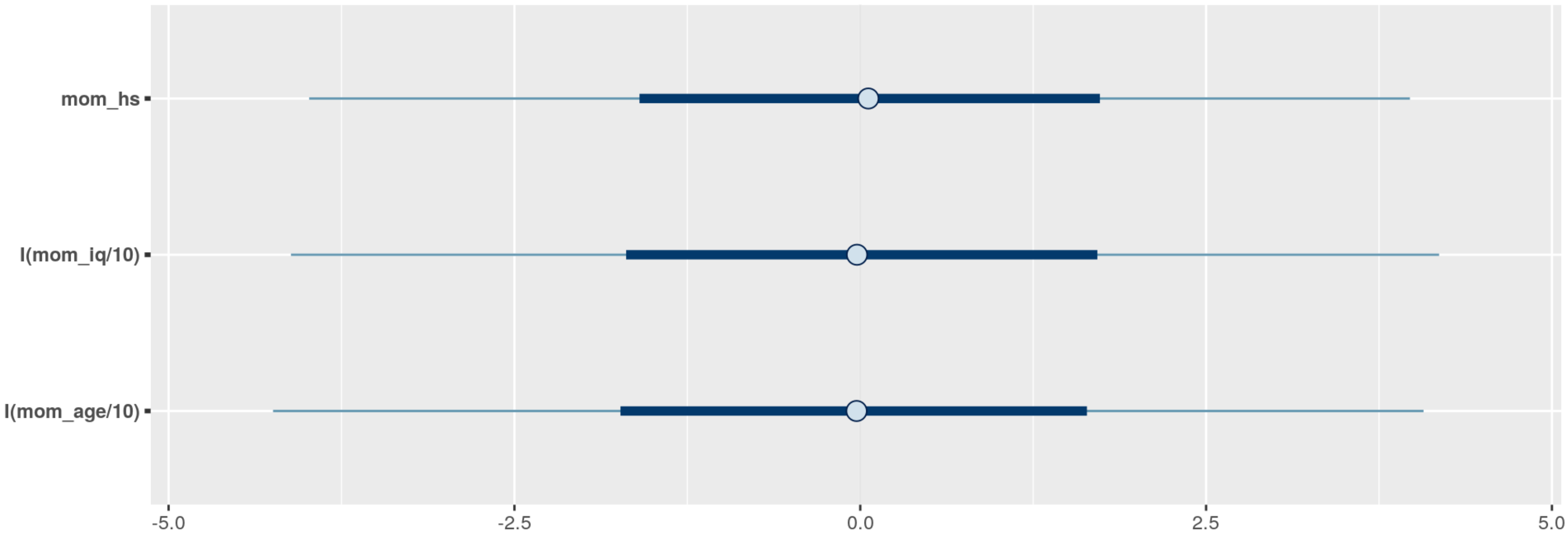
```
kid_score <- with(kidiq, t(replicate(10000, {
  alpha_ <- rnorm(1, mean = 100, sd = 15)
  beta_hs_  <- rnorm(1, mean = 0, sd = 2.5)
  beta_iq_  <- rnorm(1, mean = 0, sd = 2.5)
  beta_age_ <- rnorm(1, mean = 0, sd = 2.5)
  mu_ <- alpha_ + beta_hs_ * mom_hs + beta_iq_ * mom_iq + beta_age_ * mom_age

  sigma_ <- rexp(1, rate = 1 / 15)
  epsilon_ <- rnorm(n = length(mu_), mean = 0, sd = sigma_)
  mu_ + epsilon_
})))
summary(kid_score[ , 1]) # predictive distribution for first 3 year old (much too wide)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -992.8  -107.4   102.8   104.4   312.4  1129.2
```

Drawing from the Prior in rstanarm

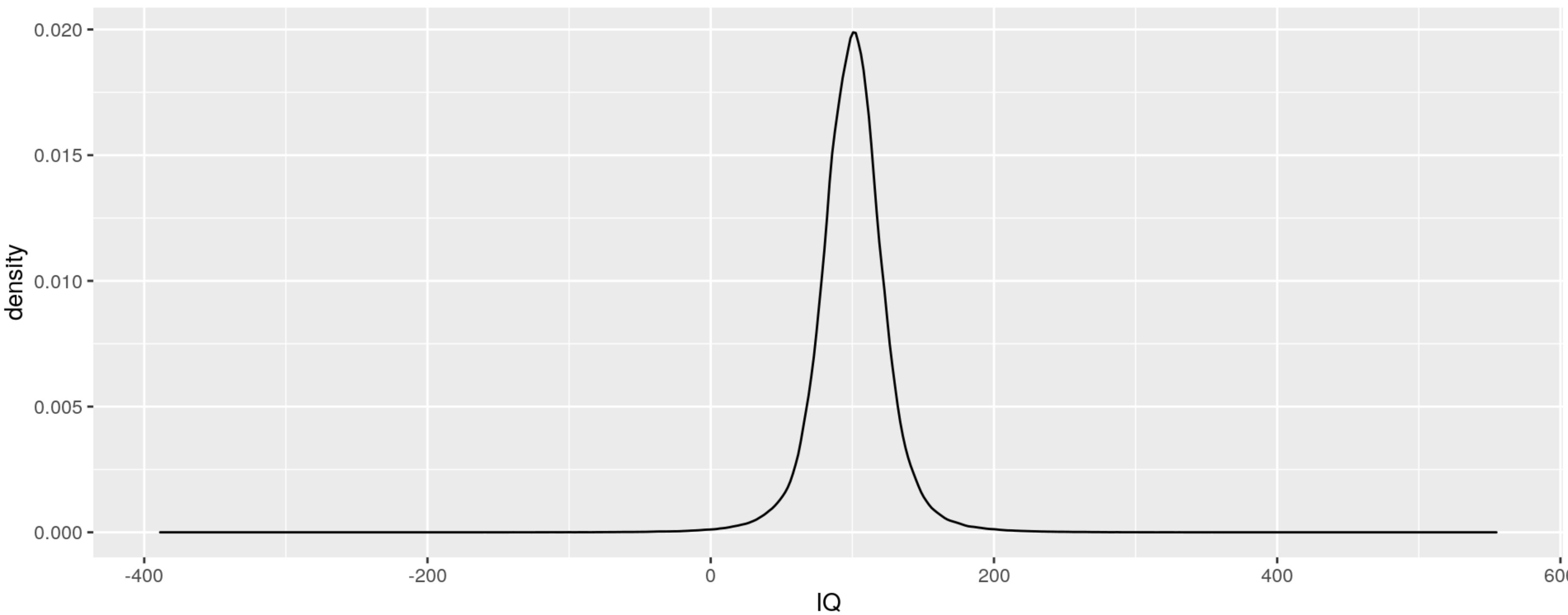
```
priors <- stan_glm(kid_score ~ mom_hs + I(mom_iq / 10) + I(mom_age / 10),
                   data = kidiq, family = gaussian(), prior_PD = TRUE,
                   prior_intercept = normal(location = 100, scale = 15),
                   prior = normal(), prior_aux = exponential(rate = 1 / 15))
```

```
plot(priors, regex_pars = "mom") # include only mom parameters
```



Prior Predictive Distribution in rstanarm

```
prior_PD <- posterior_predict(priors) # actually prior predictions
ggplot(tibble(IQ = c(prior_PD))) + geom_density(aes(x = IQ)) # tails a bit too long
```



Drawing from the Posterior Distribution

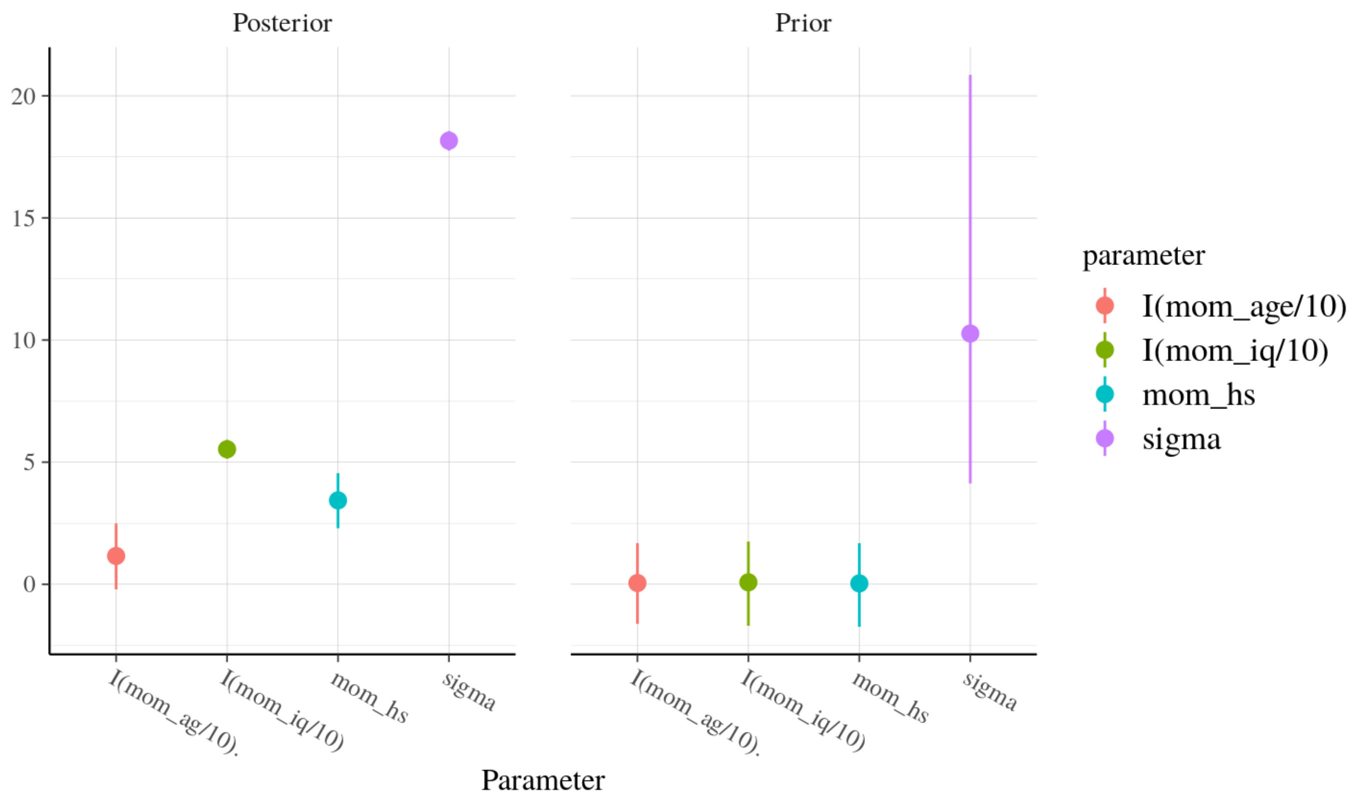
```
post <- update(priors, prior_PD = FALSE)
```

```
summary(post)
```

```
...
##                  mean   sd   10%   50%   90%
## (Intercept)   26.3    7.1 17.2  26.4  35.1
## mom_hs         3.4    1.7  1.3   3.4   5.5
## I(mom_iq/10)   5.5    0.6  4.8   5.5   6.3
## I(mom_age/10)  1.1    2.0 -1.4   1.2   3.7
## sigma         18.2    0.6 17.4  18.2  19.0
##
## Fit Diagnostics:
##             mean   sd   10%   50%   90%
## mean_PPD 86.8    1.2 85.3  86.8  88.4
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variabl
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.1  1.0  4847
## mom_hs        0.0  1.0  5832
## I(mom_iq/10)  0.0  1.0  4829
```

Posterior vs. Prior

```
posterior_vs_prior(post, prob = 0.5, regex_pars = "^[^(]") # excludes (Intercept)
```



ShinyStan

- ShinyStan can be launched on an object produced by rstanarm via

```
launch_shinystan(post)
```

- A webapp will open in your default web browser that helps you visualize the posterior distribution and diagnose problems

Linear Models with Nonlinear Predictors

- `stan_lm` and `stan_glm` (with `family = gaussian`) only require that μ be a linear function of the coefficients but allow μ to be a nonlinear function of x
- For example, you can utilize polynomials or a “restricted cubic spine” function

```
post <- stan_lm(kid_score ~ mom_hs + rms::rcs(mom_iq) + rms::rcs(mom_age),
                data = kidiq, prior = R2(0.25, what = "mode"),
                prior_intercept = normal(location = 100, scale = 15))
print(post)
```

```
...
##                              Median MAD_SD
## (Intercept)                   85.8   14.6
## mom_hs                         5.3    2.2
## rms::rcs(mom_iq)mom_iq         0.8    0.3
## rms::rcs(mom_iq)mom_iq'        0.1    2.8
## rms::rcs(mom_iq)mom_iq''      -2.8    9.2
## rms::rcs(mom_iq)mom_iq'''      4.8   10.5
## rms::rcs(mom_age)mom_age      -4.1    1.2
## rms::rcs(mom_age)mom_age'     24.5   10.1
## rms::rcs(mom_age)mom_age''  -65.0   36.4
## rms::rcs(mom_age)mom_age'''  64.1   77.3
##
## Auxiliary parameter(s):
##               Median MAD_SD
## R2             0.2    0.0
## log-fit_ratio  0.0    0.0
## sigma         18.1    0.6
...
```

Don’t (Mis)Interpret; Plot your Posterior Beliefs

```
beta_age <- as.matrix(post)[ , 3:6] # 4000 x 4
X <- rms::rcs(sort(kidiq$mom_iq))   # 434 x 4
effect_iq <- beta_age %*% t(X)      # 4000 x 434
quantiles <- apply(effect_iq, MARGIN = 2, FUN = quantile, probs = c(.1, .25, .5, .75, .9))
matplot(x = sort(kidiq$mom_iq), y = t(quantiles), type = "l",
        xlab = "Mom's IQ (when kid is born)", ylab = "Kid's IQ") # 1 color per quantile
```