

# Introduction to Bayesian linear regression with brms

---

Stefano Coretta

18/01/2020

# Installation

Safe method:

- **Install Rstan first:** <https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started> (see *Installation of Rstan*, *Checking the C++ Toolchain*, and *Configuration of the C++ Toolchain*).
  - Note that details in *Checking the C++ Toolchain* differ depending on your OS.
- **Install brms:** <https://github.com/paul-buerkner/brms#how-do-i-install-brms>.

# Road map

- Basic concepts
- Model fitting with brms
- Bayesian inference
- BYOD clinic

# Learning outcomes

- Understand the basic concepts behind Bayesian statistics and how this differs from frequentist statistics.
- Be able to set up and fit a Bayesian linear regression model using brms.
- Understand Bayesian inference methods.

# BASIC CONCEPTS

# Random variables

- We have a question about the world, so we collect data (sample from a population).
  - $y = (y_1, y_2, y_3, y_4, \dots, y_n)$
- We want to know how the data (the sample  $y$ ) was generated.
- In probability theory, data is generated by a random variable  $Y$ .

# Random variables

- $Y$  is a variable whose value is generated by a random event.
- $Y$  is uncertain.
  - We can describe  $Y$  as a probability distribution.

# Random variables

- Probability distribution.
  - A list of the values a random variable could take on along with their corresponding probability.
- Probability distributions can be expressed by a set of parameters  $\Theta = (\theta_1, \dots, \theta_n)$ .
- Some probability distributions:
  - *Normal*( $\mu, \sigma$ ),
  - *Binomial*( $n, p$ ),
  - *Poisson*( $\lambda$ )



# Frequentist vs Bayesian view

- Parameters:  $\mu, \sigma, p, \lambda, \dots$
- Frequentist view:
  - The parameters are **fixed** (they are unknown but certain).
  - They take on a specific value.
- Bayesian view:
  - The parameters are **random variables** (they are unknown and uncertain).
  - We describe each parameter as a probability distribution, expressed by a set of **hyperparameters**.

$$vot_i \sim Normal(\mu, \sigma)$$

# Continuous random variable

- We want to estimate the parameters  $\mu$  and  $\sigma$  from the data.
  - We describe the parameters using probability distributions (defined by hyperparameters).

## Continuous random variable

$$vot_i \sim Normal(\mu, \sigma)$$

$$\mu \sim Normal(\mu_1, \sigma_1)$$

$$\sigma \sim HalfCauchy(x_0, \gamma)$$

# Priors

- We can incorporate previous knowledge (belief) about the parameters using **priors** (*prior probability distributions*).
  - We specify the hyperparameters  $\mu_1$ ,  $\sigma_1$ ,  $x_0$ ,  $\gamma$  of the prior probability distributions.
- Priors are chosen based on expert knowledge, previous studies, pilot data...
  - Priors must **not** be chosen based on the data to be analysed.

## Bayesian belief update

*prior probability*  $\times$  *evidence* (data)

# Bayesian belief update

<https://nanx.shinyapps.io/conjugate-normal-umkv/>

# MODEL FITTING



- Voice Onset Time (VOT): Time difference (ms) between the release of a stop and the onset of voicing (vocal fold vibration).
- Toy study of Italian VOT in stops.

$$vot_i \sim Normal(\mu, \sigma)$$

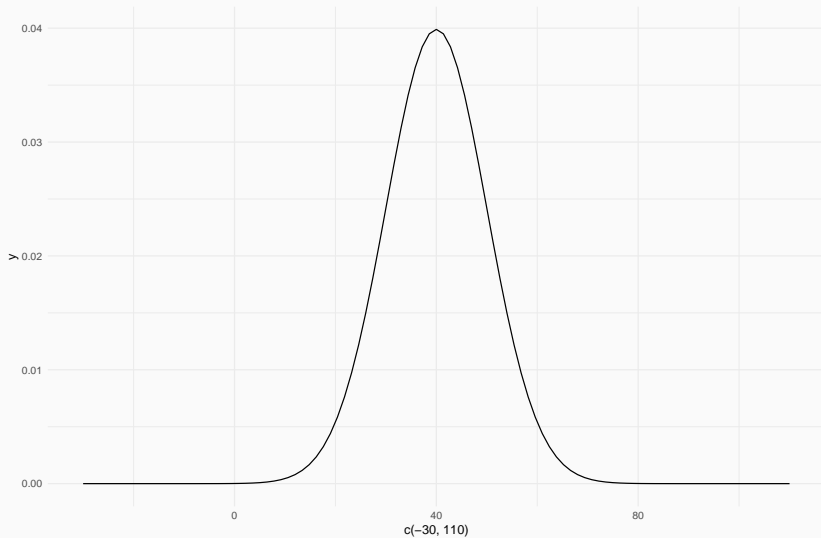
- Previous literature on VOT in Italian (Esposito, 2002; Stevens & Hajek, 2010) report VOT values for voiceless stops in the range of 20–60 ms.
- We can include our knowledge with a prior for  $\mu$ .
  - *Normal*(40, 10).
  - This is a somewhat strongly informative prior.

```
ggplot() +  
  aes(x = c(-30, 110)) +  
  stat_function(fun = dnorm, n = 101, args = list(40,  
  labs(title = "Normal (Gaussian) distribution", subti-
```

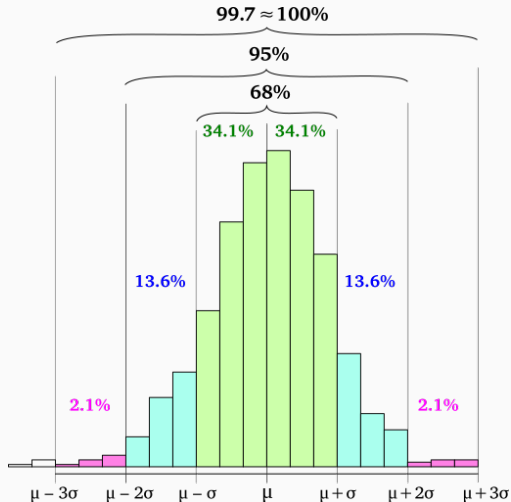
# Italian VOT

Normal (Gaussian) distribution

mean = 40, SD = 10



# Normal prior



Melikamp, [https://commons.wikimedia.org/wiki/File:Empirical\\_rule\\_histogram.svg](https://commons.wikimedia.org/wiki/File:Empirical_rule_histogram.svg) (CC BY-SA 4.0)

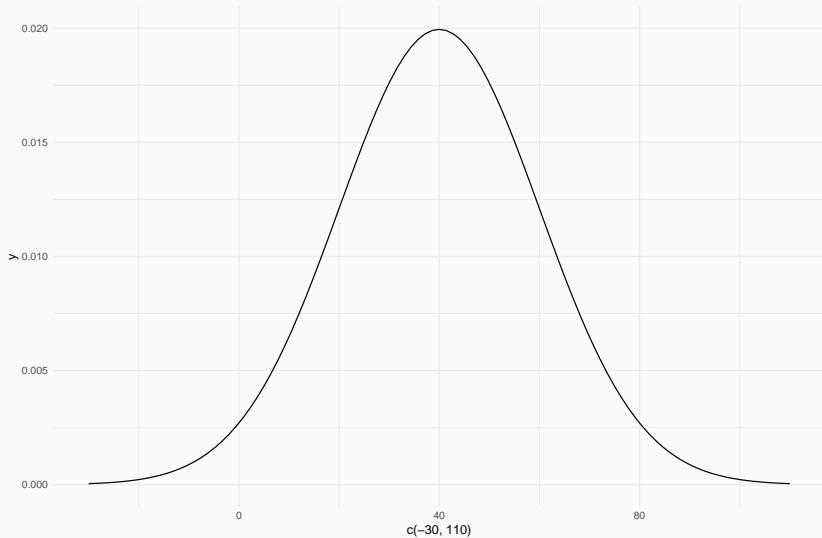
# Italian VOT

```
ggplot() +  
  aes(x = c(-30, 110)) +  
  stat_function(fun = dnorm, n = 101, args = list(40, 20)) +  
  labs(title = "Normal (Gaussian) distribution", subtitle = "mean = 40, SD = 20")
```

# Italian VOT

Normal (Gaussian) distribution

mean = 40, SD = 20



$$vot_i \sim Normal(\mu, \sigma)$$

$$\mu \sim Normal(40, 10)$$

What about  $\sigma$ ?



$$vot_i \sim Normal(\mu, \sigma)$$

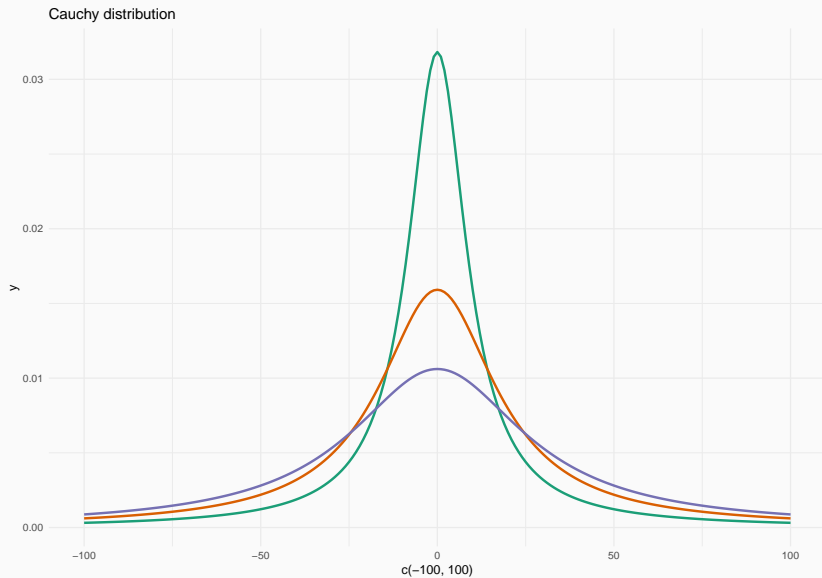
$$\mu \sim Normal(40, 10)$$

$$\sigma \sim HalfCauchy(x_0, \gamma)$$

# Cauchy prior

```
ggplot() +  
  aes(x = c(-100, 100)) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 10), colour = "#1b9e77", size = 1) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 20), colour = "#d95f02", size = 1) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 30), colour = "#7570b3", size = 1) +  
  labs(title = "Cauchy distribution")
```

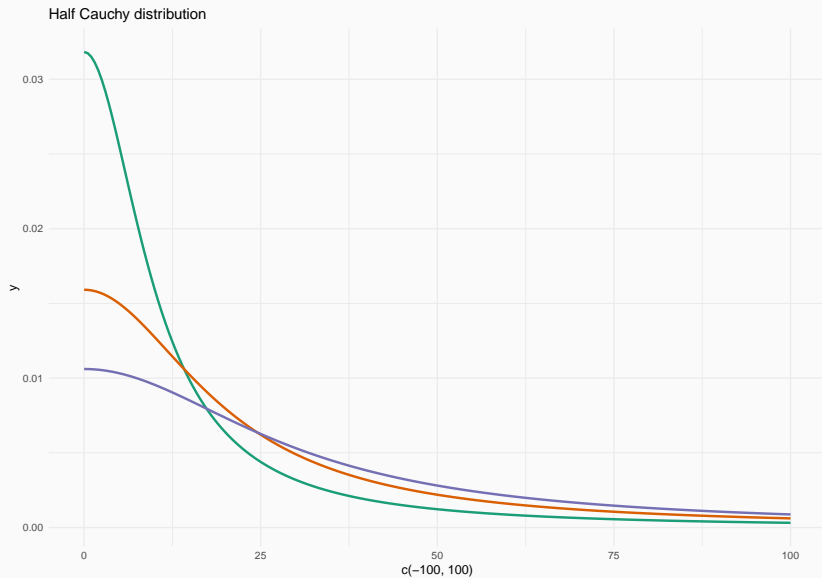
# Cauchy prior



# Cauchy prior

```
ggplot() +  
  aes(x = c(-100, 100)) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 10), colour = "#1b9e77", size = 1) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 20), colour = "#d95f02", size = 1) +  
  stat_function(fun = dcauchy, n = 201, args = list(0, 30), colour = "#7570b3", size = 1) +  
  labs(title = "Half Cauchy distribution") +  
  xlim(0, 100)
```

# Cauchy prior



$$vot_i \sim Normal(\mu, \sigma)$$

$$\mu \sim Normal(40, 10)$$

$$\sigma \sim HalfCauchy(0, 10)$$

- We have a model which incorporates (some of) our knowledge about VOT (through the priors for  $\mu$  and  $\sigma$ ).
- Now we want to obtain the **posterior distributions** of  $\mu$  and  $\sigma$ .
  - The posterior distribution is the prior distribution *conditioned* on the data.
- **brms** R package: Bayesian Regression Models using Stan (Bürkner, 2018).

- Stan (Stan Development Team, 2017).
  - Statistical programming language written in C++.
  - Fit Bayesian models (calculate posterior distributions).
- Calculation can be complex and/or impossible, so we take many samples from the data and from the possible parameter values to find the posterior distributions of the hyperparameters.
  - Markov Chain Monte Carlo (MCMC) sampling using the No-U-Turn sampler (NUTS).



- brms translates R code into Stan code.
- Stan code is run in Stan via Rstan, an R interface to Stan.
- `brm()` function from brms.
  - lme4 syntax ( $y \sim x + (1|w)$ ).
  - Creates a Stan model, which is compiled (in C++) and run.

```
library(brms)

vot1 <- brm(
  <model_formula>,
  <family>,
  <prior>,
  <data>,
  chains = 4,
  iter = 2000
)
```

```
library(brms)

vot1 <- brm(
  vot ~ 1,
  family = gaussian(),
  <prior>,
  data = ita_egg,
  chains = 4,
  iter = 2000
)
```

# Get prior

```
get_prior(  
  vot ~ 1,  
  family = gaussian(),  
  data = ita_egg  
)
```

```
##               prior      class coef group resp dpar nlpar bound  
## 1 student_t(3, 19, 14) Intercept  
## 2 student_t(3, 0, 14)      sigma
```

# Prior predictive checks

```
nsim <- 1000
nobs <- 1000

y <- matrix(rep(NA, nsim * nobs), ncol = nobs)

mu <- rnorm(nsim, 40, 10)
sigma <- rhcauchy(nsim, 10)

for (i in 1:nsim) {
  y[i,] <- rnorm(nobs, mean = mu[i], sd = sigma[i])
}
```

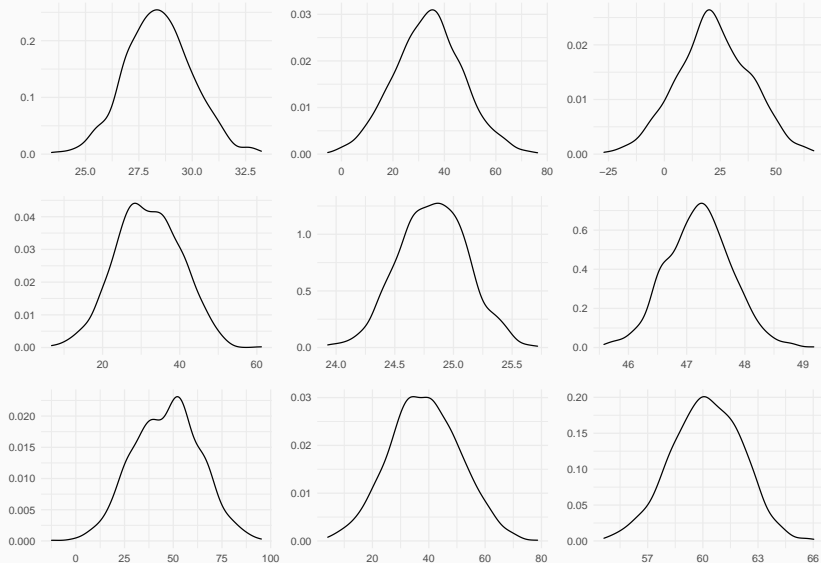
## Prior predictive checks

```
rand_sample <- sample(1:nsim, 9, replace = FALSE)
plots <- list()

j = 1

for (i in rand_sample) {
  my_data <- enframe(y[i,], name = NULL)
  plots[[j]] <- ggplot(data = my_data) +
    aes(x = value) +
    geom_density() +
    labs(x = element_blank(), y = element_blank())
  j = j + 1
}
```

# Prior predictive checks



# Set prior

```
priors <- c(  
  prior(normal(40, 10), class = Intercept),  
  prior(cauchy(0, 10), class = sigma)  
)
```



# Run the model

```
vot1 <- brm(  
  vot ~ 1,  
  family = gaussian(),  
  prior = priors,  
  data = ita_egg,  
  chains = 4,  
  iter = 2000,  
  file = "./cache/vot1"  
)
```

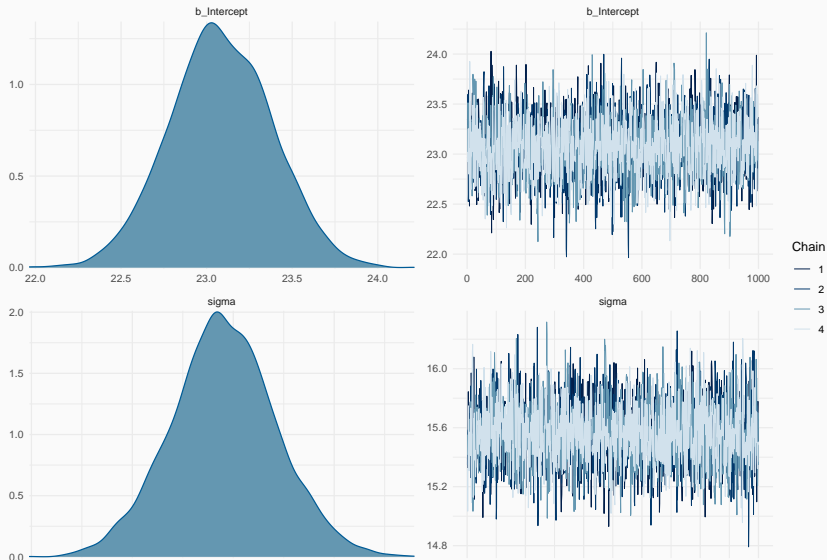
# Model summary

```
vot1

## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: vot ~ 1
## Data: ita_egg (Number of observations: 2624)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##           total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept      23.08      0.30   22.49   23.67 1.00    3184    2635
##
## Family Specific Parameters:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma      15.56      0.21   15.15   15.98 1.00    3391    2325
##
## 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).
```

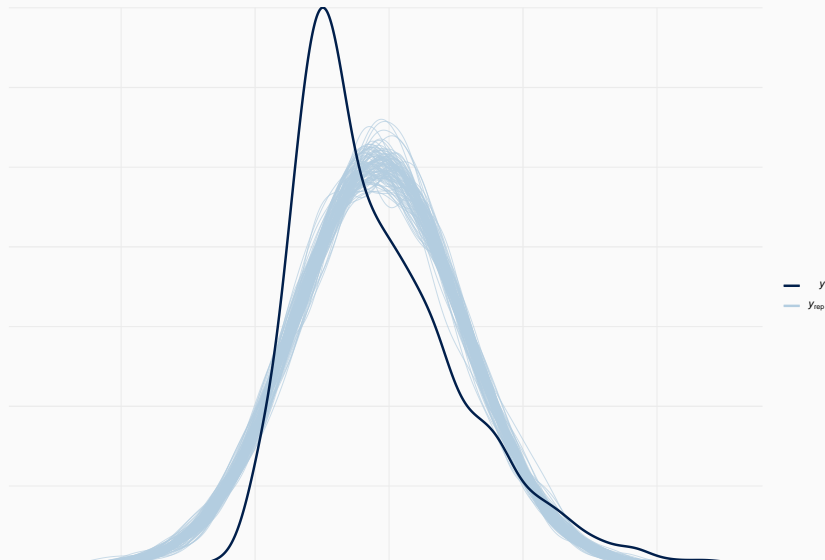
# Plot model

```
plot(vot1)
```



# Posterior predictive check

```
pp_check(vot1, nsamples = 100)
```



# Adding predictors

$$vot_i \sim Normal(\mu_i, \sigma)$$

$$\mu_i = \alpha + \beta_1 \times coronal_i + \beta_2 \times velar_i$$

$$\alpha \sim Normal(\mu_1, \sigma_1)$$

$$\beta_1 \sim Normal(\mu_2, \sigma_2)$$

$$\beta_2 \sim Normal(\mu_3, \sigma_3)$$

$$\sigma \sim HalfCauchy(x_0, \gamma)$$

## Adding predictors

$$vot_i \sim Normal(\mu_i, \sigma)$$

$$\mu_i = \alpha + \beta_1 \times coronal_i + \beta_2 \times velar_i$$

$$\alpha \sim Normal(25, 10)$$

$$\beta_1 \sim Normal(10, 10)$$

$$\beta_2 \sim Normal(20, 10)$$

$$\sigma \sim HalfCauchy(0, 10)$$

# Adding predictors

```
get_prior(  
  vot ~ 1 + c1_place,  
  family = gaussian(),  
  data = ita_egg  
)
```

```
##           prior      class      coef group resp dpar nlpar bound  
## 1                      b  
## 2                      b c1_placecoronal  
## 3                      b  c1_placevelar  
## 4 student_t(3, 19, 14) Intercept  
## 5 student_t(3, 0, 14)   sigma
```

# Adding predictors

```
priors <- c(  
  prior(normal(25, 10), class = Intercept),  
  prior(cauchy(0, 10), class = sigma),  
  prior(normal(10, 10), class = b, coef = "ci_placecoronal"),  
  prior(normal(20, 10), class = b, coef = "ci_placevelar")  
)
```



# Adding predictors

```
vot2 <- brm(  
  vot ~ 1 + c1_place,  
  family = gaussian(),  
  prior = priors,  
  data = ita_egg,  
  chains = 4,  
  iter = 2000,  
  file = "./cache/vot2"  
)
```

# Random effects

$$vot_i \sim Normal(\mu_i, \sigma)$$

$$\mu_i = \alpha + \alpha_{speaker[i]} + (\beta_1 + \beta_{1speaker[i]}) \times coronal_i + (\beta_2 + \beta_{2speaker[i]}) \times velar_i$$

$$\begin{bmatrix} \alpha_{speaker} \\ \beta_{1speaker[i]} \\ \beta_{2speaker[i]} \end{bmatrix} \sim MVNormal\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, S\right)$$

# Random effects

$$\alpha \sim \text{Normal}(25, 10)$$

$$\alpha_{speaker} \sim \text{Normal}(0, \sigma_{speaker})$$

$$\beta_1 \sim \text{Normal}(10, 10)$$

$$\beta_2 \sim \text{Normal}(20, 10)$$

$$\sigma_{\alpha speaker} \sim \text{Normal}(0, \sigma_{speaker})$$

$$\sigma_{\beta_1 speaker} \sim \text{HalfCauchy}(0, 10)$$

$$\sigma_{\beta_2 speaker} \sim \text{HalfCauchy}(0, 10)$$

$$\sigma_{speaker} \sim \text{HalfCauchy}(0, 10)$$

$$\sigma \sim \text{HalfCauchy}(0, 10)$$

# Random effects

```
get_prior(  
  vot ~ 1 + c1_place + (1 + c1_place | speaker),  
  family = gaussian(),  
  data = ita_egg  
)
```

```
##           prior      class      coef  group resp dpar nlpar bound  
## 1                b  
## 2                b c1_placecoronal  
## 3                b  c1_placevelar  
## 4      lkj(1)      cor  
## 5                cor                speaker  
## 6 student_t(3, 19, 14) Intercept  
## 7 student_t(3, 0, 14)      sd  
## 8                sd                speaker  
## 9                sd c1_placecoronal speaker  
## 10               sd  c1_placevelar speaker  
## 11               sd      Intercept speaker  
## 12 student_t(3, 0, 14)      sigma
```

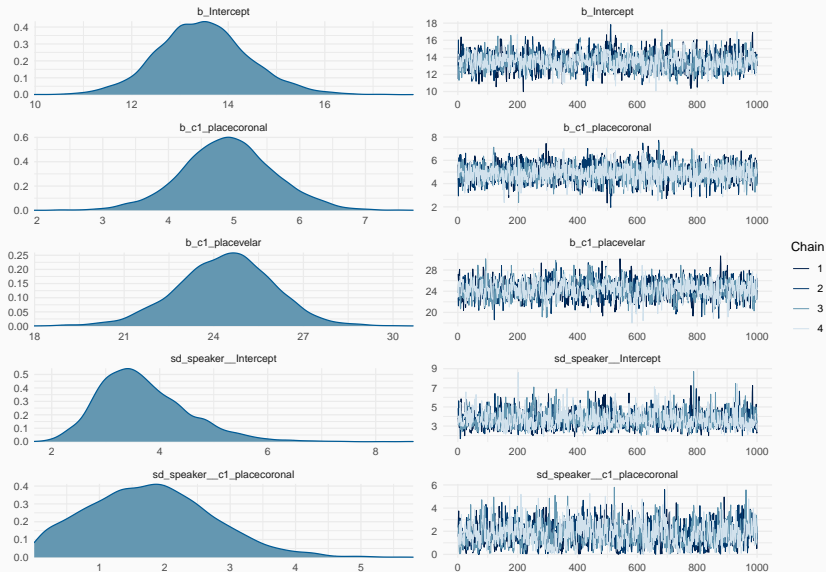
# Random effects

```
priors <- c(  
  prior(normal(40, 10), class = Intercept),  
  prior(cauchy(0, 10), class = sigma),  
  prior(normal(10, 10), class = b, coef = "ci_placecoronal"),  
  prior(normal(20, 10), class = b, coef = "ci_placevelar"),  
  prior(normal(0, 25), class = sd),  
  prior(lkj(2), class = cor)  
)
```

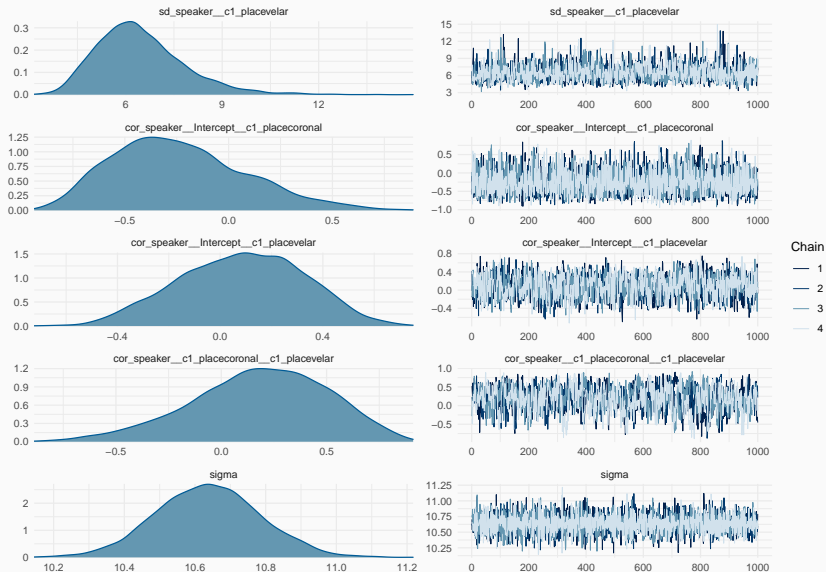
# Random effects

```
vot3 <- brm(  
  vot ~ 1 + c1_place + (1 + c1_place | speaker),  
  family = gaussian(),  
  prior = priors,  
  data = ita_egg,  
  chains = 4,  
  iter = 2000,  
  file = "./cache/vot3"  
)
```

# Random effects



# Random effects





# Random effects

```
cat(capture.output(summary(vot3))[1:16], sep = "\n")

## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: vot ~ 1 + c1_place + (1 + c1_place | speaker)
## Data: ita_egg (Number of observations: 2624)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Group-Level Effects:
## ~speaker (Number of levels: 18)
##
##          Estimate Est.Error 1-95% CI u-95% CI Rhat
## sd(Intercept)          3.70    0.83    2.38    5.57 1.00
## sd(c1_placecoronal)     1.78    0.94    0.16    3.77 1.00
## sd(c1_placevelar)       6.46    1.38    4.31    9.70 1.00
## cor(Intercept,c1_placecoronal) -0.24    0.32   -0.75    0.47 1.00
## cor(Intercept,c1_placevelar)   0.08    0.24   -0.39    0.53 1.00
## cor(c1_placecoronal,c1_placevelar) 0.18    0.32   -0.51    0.74 1.01
```

# Random effects

```
cat(capture.output(summary(vot3))[18:30], sep = "\n")
```

```
## sd(Intercept)                1379    1918
## sd(c1_placecoronal)          729    1246
## sd(c1_placevelar)            1375    1881
## cor(Intercept,c1_placecoronal) 2747    2459
## cor(Intercept,c1_placevelar)  1274    2002
## cor(c1_placecoronal,c1_placevelar) 582    862
##
## Population-Level Effects:
##               Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept      13.48      0.95   11.64   15.46 1.00    1081    1672
## c1_placecoronal  4.91      0.70    3.50    6.32 1.00    2610    2153
## c1_placevelar   24.38     1.61   21.14   27.40 1.00    1218    1547
```

# Binomial logistic regression

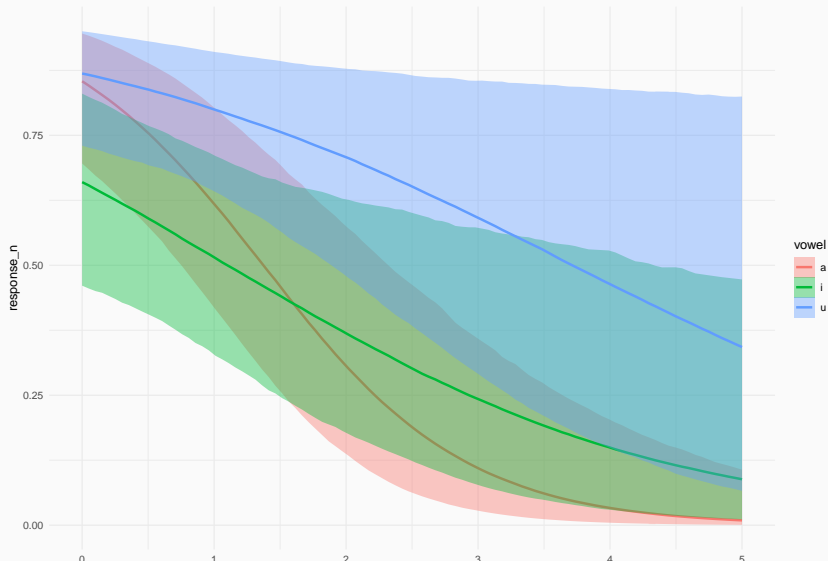
```
priors <- c(  
  prior(student_t(3, 0, 2), class = Intercept),  
  prior(student_t(3, 0, 2), class = b),  
  prior(cauchy(0, 1), class = sd),  
  prior(lkj(2), class = cor)  
)
```

# Binomial logistic regression

```
burst1 <- brm(  
  response_n ~  
    burst *  
    vowel +  
    (1+burst|participant),  
  data = burst,  
  prior = priors,  
  family = bernoulli,  
  file = "./cache/burst1",  
  control = list(adapt_delta = 0.999)  
)
```

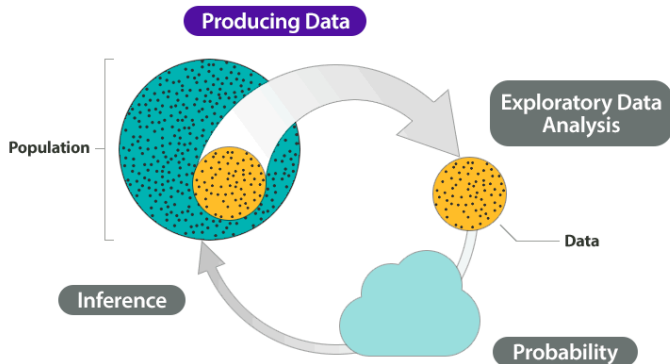
# Binomial logistic regression

```
conditional_effects(burst1, effects = "burst:vowel")
```



# BAYESIAN INFERENCE

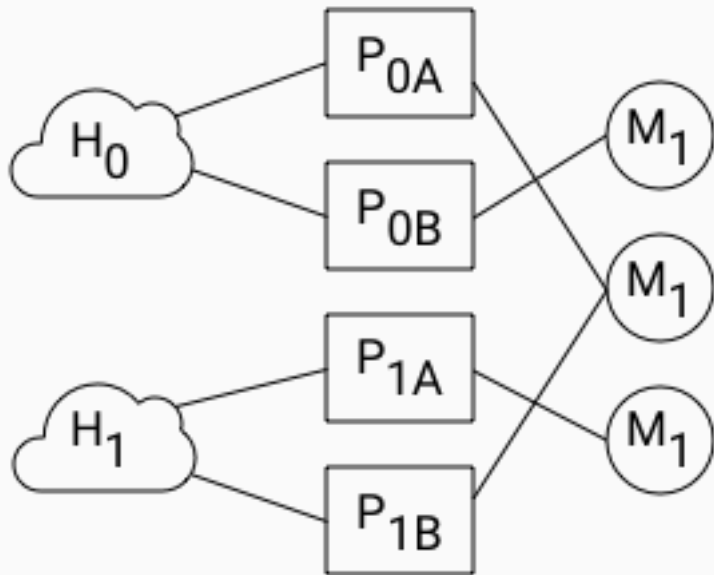
# Inference



Open Learning Initiative,

<https://oli.cmu.edu/courses/concepts-of-statistics/> (CC BY-NC-SA 4.0)

# Inference





# Inference

- We want to know two things:
  - If there is **evidence for our hypothesis  $H$**  (or for the value of the parameter  $\theta$ ), and
  - What the **strength** of the evidence is.

- **Inferential statistics.**
- We test  $H$  against empirical data (hypothesis testing).
  - It is important to decide in advance the details of the analysis (model and prior specification among other things).
- Inference is ultimately a long-term endeavour (via accumulation of knowledge).

- Three ways of doing inference (hypothesis testing) with Bayesian statistics:
  - Inference from the **posterior**.
  - Inference using a **Region Of Practical Equivalence (ROPE)**.
  - Inference using the **Bayes factor**.

# Inference from the posterior

- **H:** Condition B decreases reaction times relative to Condition A.
  - You have chosen a prior which appropriately conveys the content of this H.
- **Posterior:** Condition B 95% CI = [-80, -15] ms.
- **Inference:** The posterior suggests that Condition B decreases reaction times by 15 to 80 ms at 95% confidence.

# Inference from the posterior

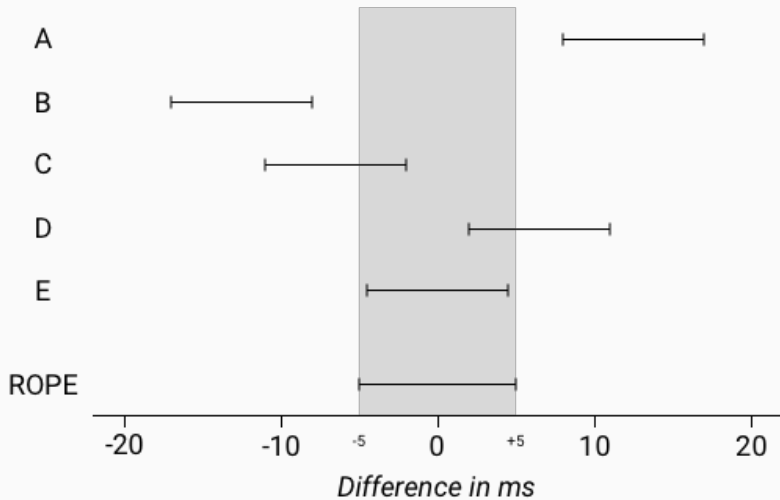
- **H:** Condition B decreases reaction times relative to Condition A *by 100 ms*.
  - You have chosen a prior which appropriately conveys the content of this H.
- **Posterior:** Condition B 95% CI = [-80, -15] ms.
- **Inference:** The posterior suggests that Condition B decreases reaction times by a smaller amount than expected from H (15 to 80 ms at 95% confidence).

# Inference with a ROPE

## H0 vs H1

- H1 states that Condition B increases segment duration (alternative hypothesis), while H0 states that Condition B does not increase segment duration (null hypothesis, null effect).
  - $H_1 : \beta > 0$
  - $H_0 : \beta = 0$
- Region of Practical Equivalence (ROPE):
  - Define a region around  $\beta = 0$  that practically corresponds to a null effect.
    - For example:  $[-5, +5]$  ms ( $-5 \geq \beta \leq +5 = \text{null effect}$ ).
    - This ROPE has a width of 10 ms.
  - Choose a minimal sample size (ideally based on prospective power analyses).

# Inference with a ROPE



# Bayes Factor

The Bayes factor is the ratio of the likelihood of  $H_1$  to the likelihood of  $H_2$ .

$$BF_{12} = \mathcal{L}(H_1)/\mathcal{L}(H_2)$$



# Bayes Factor

BF	$p(M1 D)$	evidence
1–3	0.5–0.75	weak
3–20	0.75–0.95	positive
20–150	0.95–0.99	strong
$> 150$	$> 0.99$	very strong

# Bayes Factor

```
vot3_bf <- brm(  
  vot ~ 1 + c1_place + (1 + c1_place | speaker),  
  family = gaussian(),  
  prior = priors,  
  data = ita_egg,  
  chains = 4,  
  iter = 20000,  
  file = "./cache/vot3_bf",  
  save_all_pars = TRUE  
)
```

```
vot3_bf_null <- brm(  
  vot ~ 1 + c1_place + (1 | speaker),  
  family = gaussian(),
```

# Bayes Factor

```
bayes_factor(vot3_bf, vot3_bf_null)
```

# Sensitivity analysis

# References

---

- Bürkner, Paul-Christian. 2018. Advanced Bayesian multilevel modeling with the R package brms. *The R Journal* 10(1). 395–411. doi:10.32614/RJ-2018-017.
- Esposito, Anna. 2002. On vowel height and consonantal voicing effects: Data from Italian. *Phonetica* 59(4). 197–231. doi:10.1159/000068347.
- Stan Development Team. 2017. Stan: A C++ library for probability and sampling, version 2.14.0. <http://mc-stan.org/>.

Stevens, Mary & John Hajek. 2010. Post-aspiration in standard Italian: some first cross-regional acoustic evidence. Paper presented at Interspeech, 26-30 September 2010, Makuhari, Chiba, Japan.