Introduction to Bayesian linear regression with brms

Stefano Coretta

18/01/2020

```
## Warning: Missing column names filled in: 'X46' [46]
## Warning: Missing column names filled in: 'X50' [50]
```

```
## Warning: Missing column names filled in: 'X50' [50]
```

```
## Warning: Missing column names filled in: 'X50' [50]
```

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 in Checking the C++ Toolchain details differ depending on OS.
  - Install brms: https://github.com/paul-buerkner/brms#how-do-i-install-brms.

Random variables

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

Random variables

- Y is uncertain.
  - We can describe Y as a probability distribution, expressed by a set of parameters $\Theta = (\theta_1, ..., \theta_n)$.
- Probability distributions:
  - $Normal(\mu, \sigma)$,
  - $Binomial(n, p)$,
  - ...

Random variables

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

$$voiced_i \sim Bernoulli(p)$$

$$DoubleDative_i \sim Poisson(\lambda)$$

Frequentist vs Bayesian view

- Parameters: μ, σ, p, λ, ...
- 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 unkown and uncertain).
  - We describe each parameter as a probability distribution, expressed by a set of **hyperparameters**.

Continous random variable

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

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

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

Bayes' Theorem

$$P(\theta \mid d) = \frac{P(d \mid \theta)\,P(\theta)}{P(d)}$$

Bayes' Theorem

$$posterior\ probability = \frac{likelihood \times prior}{marginal\ likelihood}$$

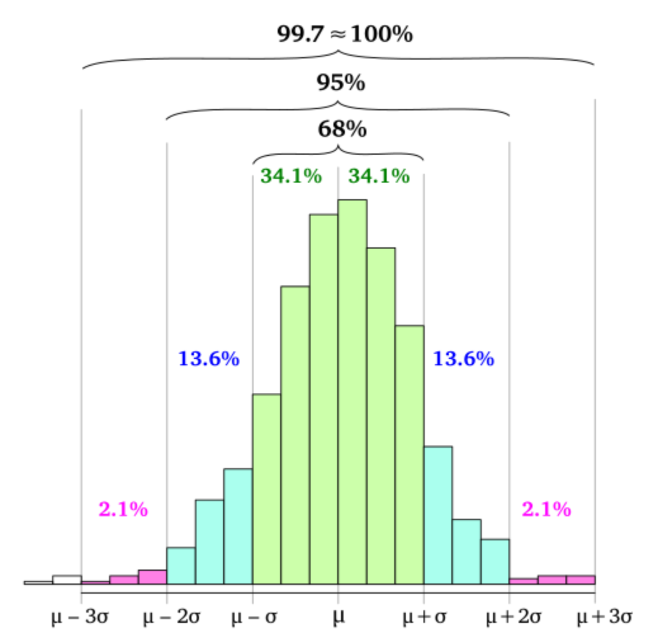
Priors

- We can incorporate previous knowledge about the hyperparameters as **priors** (prior distributions).
- Priors are chosen based on expert knowledge, previous studies, pilot data...
    - Priors must **not** be chosen based on the data to be analysed.

Priors

- Informative and weakly informative priors.
- Uninformative or diffuse priors.
  - Uniform distribution.
- Regularising priors.

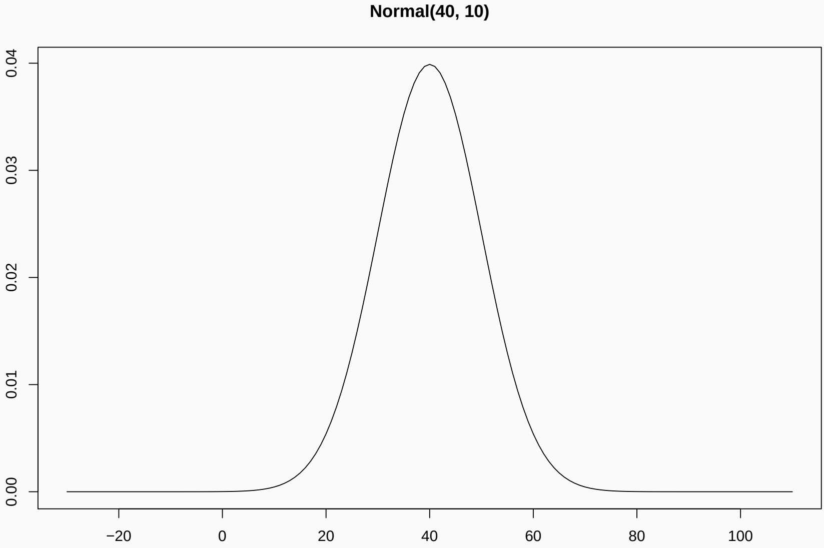
Normal prior



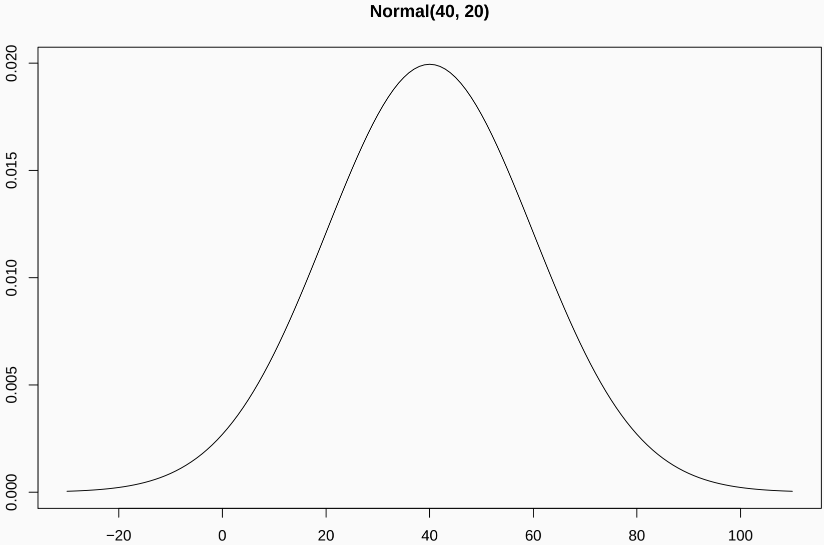
Italian VOT

- 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 express this knowledge with the prior $Normal(40, 10)$.
    - This is a somewhat strongly informative prior.

Italian VOT



Italian VOT



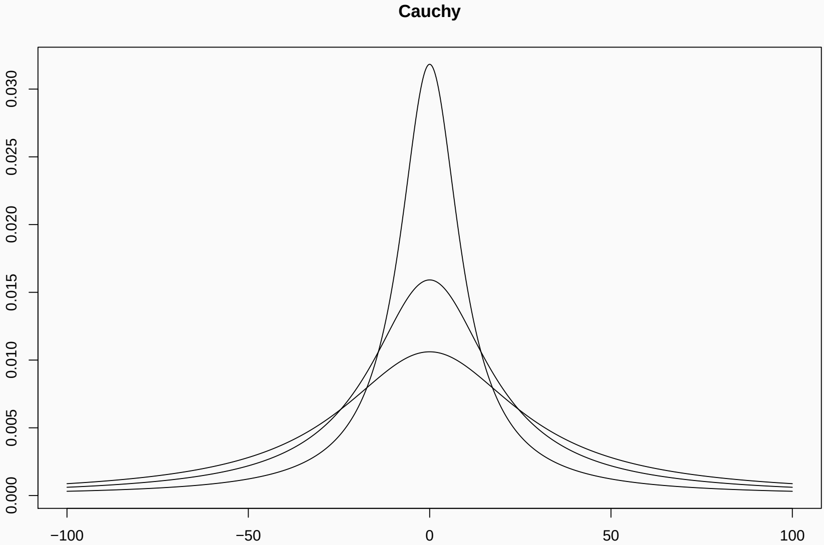
Italian VOT

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

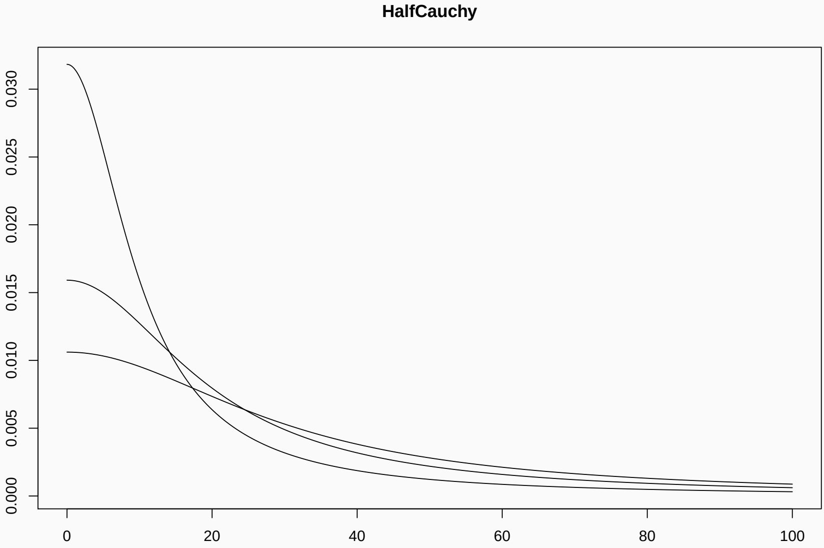
$$\mu \sim Normal(40, 10)$$

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

Cauchy prior



Cauchy prior



Italian VOT

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

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

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

Italian VOT

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

brms

- Stan (Stan Development Team, 2017).
  - Statistical programming language written in C++ for fitting 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 is an interface between R and Stan.
- `brm()` function from brms.
  - lme4 syntax (`y ~ x + (1|w)`).
  - Creates a Stan model, which is compiled and run.

brms

```r
library(brms)

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

brms

```r
library(brms)

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

Get prior

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

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

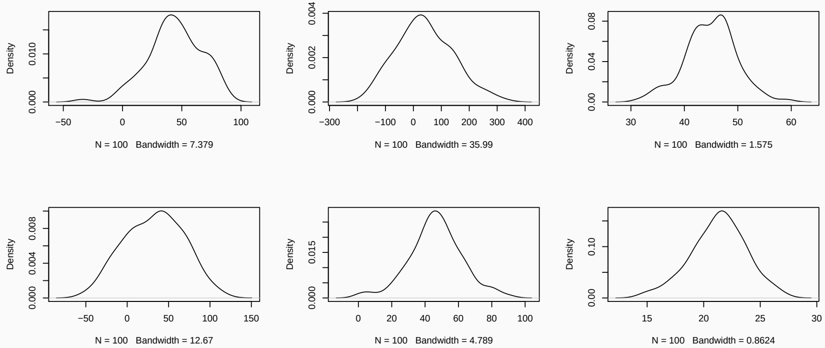
Set prior

Prior predictive checks

```r
nsim <- 1000
nobs <- 100

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

```
op <- par(mfrow = c(3, 3))
rand_sample <- sample(1:nsim, 9)

for (i in rand_sample) {
  plot(density(y[i,]), main = "")
}
```



Run the model

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

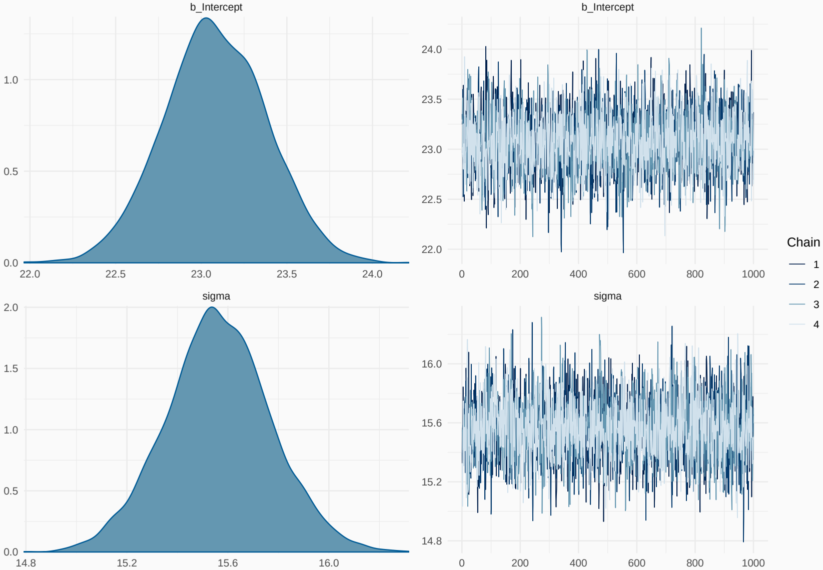
Model summary

```
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;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk
## Intercept    23.08      0.30    22.49    23.67 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sigma    15.56      0.21    15.15    15.98 1.00     3391
```

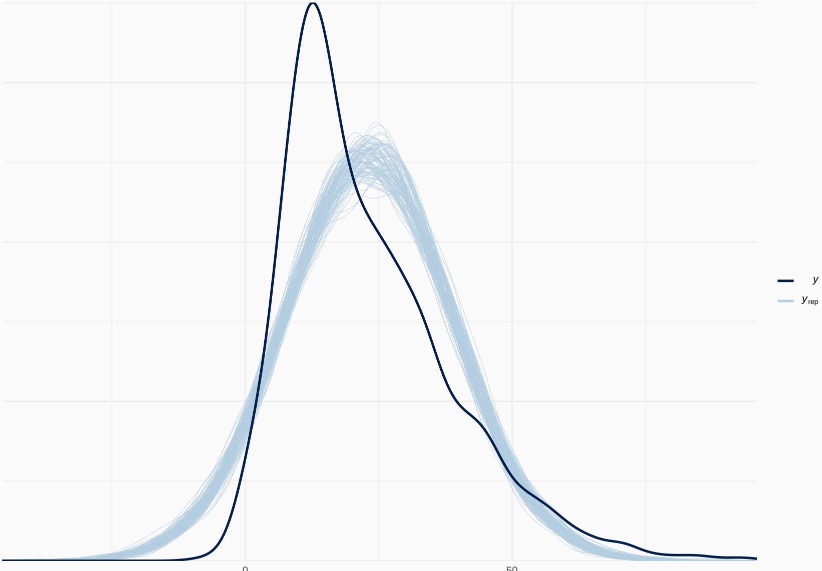
Plot model

```
plot(vot1)
```



Posterior predictive check

```r
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

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

```
##                  prior     class            coef group r
## 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 = "c1_placecoronal"
  prior(normal(20, 10), class = b, coef = "c1_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

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

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

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

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

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

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

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

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

$$\sigma_{\beta 1 speaker} \sim HalfCauchy(0, 10)$$

Random effects

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

```
##                   prior     class              coef   grou
## 1                               b
## 2                               b c1_placecoronal
## 3                               b   c1_placevelar
## 4                lkj(1)       cor
## 5                             cor                 speake
## 6  student_t(3, 19, 14) Intercept
## 7   student_t(3, 0, 14)        sd
## 8                              sd                 speake
## 9                              sd c1_placecoronal speake
```

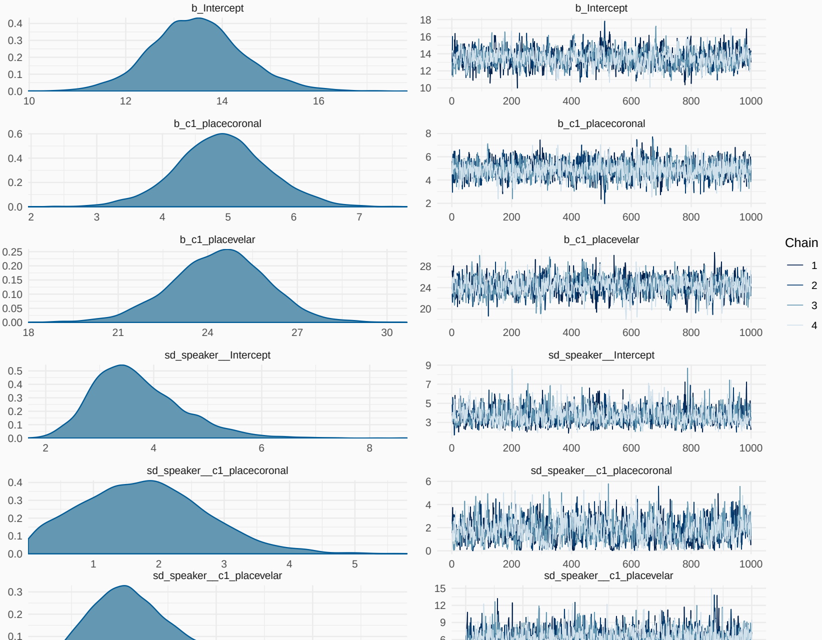
Random effects

```
priors <- c(
  prior(normal(40, 10), class = Intercept),
  prior(cauchy(0, 10), class = sigma),
  prior(normal(10, 10), class = b, coef = "c1_placecoronal"
  prior(normal(20, 10), class = b, coef = "c1_placevelar"),
  prior(normal(0, 25), class = sd),
  prior(lkj(2), class = cor)
)
```

Random effects

```r
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

```
vot3
```

```
##  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;
##          total post-warmup samples = 4000
##
## Group-Level Effects:
## ~speaker (Number of levels: 18)
##                                     Estimate Est.Error l-
## sd(Intercept)                           3.70      0.83
## sd(c1_placecoronal)                     1.78      0.94
## sd(c1_placevelar)                       6.46      1.38
## cor(Intercept,c1_placecoronal)         -0.24      0.32
```

Binomial logistic regression

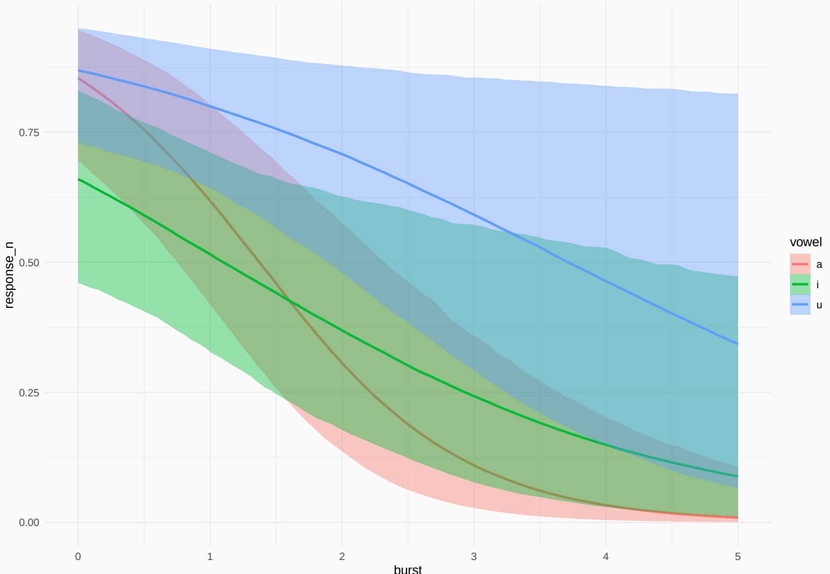
```
priors <- c(
  prior(student_t(3, 0, 2), class = Intercept),
  prior(student_t(3, 0, 2), class = b),
  prior(cauchy(0, 1), class = sd),
  prior(lkj(2), class = cor)
)
```

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")
```



Inference

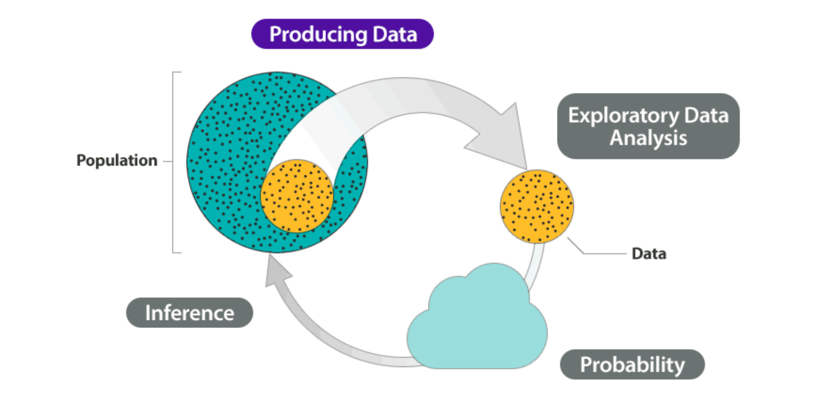


Figure 2: statistical inference

Open Learning Initiative, https://oli.cmu.edu/courses/concepts-of-statistics/ (CC BY-NC-SA 4.0)

Inference

- We want to know two things:
    - If there is **evidence for our hypothesis H** (or for the value of the parameter θ), and
    - What the **strength** of the evidence is.
- This is the aim of 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 endaveour (via accumulation of knowledge).

Inference

- 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.
- Posterior: 95% CI = [-80, -15] ms.
- Inference: The posterior suggests that Condition B decreases reaction times by 15 to 80 ms at 95% confidence.

Inference with a ROPE

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.