Bayesian Analysis using Stan and brms

Xiaochuan Li
2022-06-02

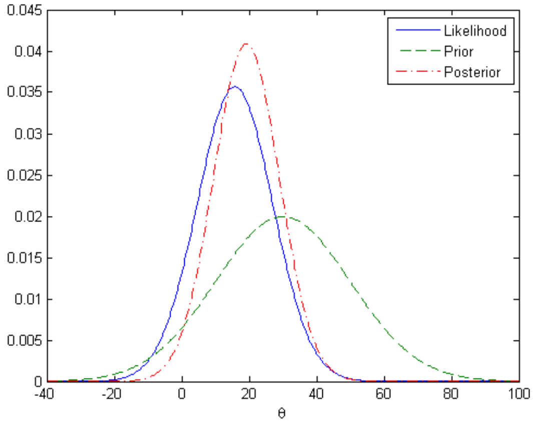
Agenda

- Bayesian Analysis
- Software
- brms
- Examples

Frequentist vs. Bayesian analysis

- Frequentist Statistics
  - highest probability of seeing the data given (conditional on) the underlying model
- Bayesian Statistics
  - Which model, or set of parameter values, are most certain to be the true model
  - Bayes' Theorem

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$



Why use Bayesain analysis

- Prior Information
- Hierarchical Models
- more interpretable confidence intervals
- Limited Data
    - Prior distribution over the parameters can act as a regularization to prevent unlikely extreme values

Software

R interface

- lme4, rethinking::quap - Quadratic approximation
- BUG, JAGS - Gibbs sampling
- STAN - Hamiltonian Sampling

Python interface

- PyMc3 - Hamiltonian Monte Carlo
- Pytorch::Pyro - Bayesian statistical models and Neural Networks

Stan

Linear model

$$log(y_i) \sim N(b_0 + b_1 log(x_i), \sigma^2)$$

data block

```
data {
int<lower=1> N; // number of observations
vector[N] log_gest; // log gestational age
vector[N] log_weight; // log birth weight
}
```

parameters block

```
parameters {
vector[2] beta; // coefs
real<lower=0> sigma; // error sd for Gaussian likelihood
}
```

model block

```
model {
// Log-likelihood
target += normal_lpdf(log_weight | beta[1] + beta[2] * log_gest, sigma
// Log-priors
target += normal_lpdf(sigma | 0, 1)
+ normal_lpdf(beta | 0, 1);
}
```

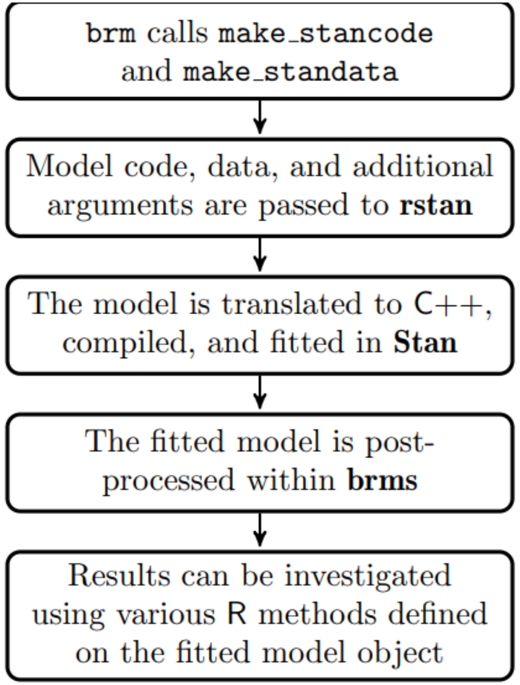
generated quantities block

```
generated quantities {
vector[N] log_lik; // pointwise log-likelihood for LOO
vector[N] log_weight_rep; // replications from posterior predictive di
for (n in 1:N) {
real log_weight_hat_n = beta[1] + beta[2] * log_gest[n];
log_lik[n] = normal_lpdf(log_weight[n] | log_weight_hat_n, sigma);
```

brms

brms provides an interface to fit Bayesian generalized (non-)linear multivariate multilevel models using Stan.

- brms ---> Stan ---> C++



`brm` calls `make_stancode` and `make_standata`

Model code, data, and additional arguments are passed to **rstan**

The model is translated to C++, compiled, and fitted in **Stan**

The fitted model is post-processed within **brms**

Results can be investigated using various R methods defined on the fitted model object

High level description of the model fitting procedure used in **brms**.

- Domino environment: DAD_STAN

brms formula

```
fit1 <- brm(formula = time | cens(censored) ~ age * sex + disease + (1 + age|patient),
data = kidney, family = lognormal(),
prior = c(set_prior("normal(0,5)", class = "b"),
set_prior("cauchy(0,2)", class = "sd"),
set_prior("lkj(2)", class = "cor")),
warmup = 1000, iter = 2000, chains = 4,
control = list(adapt_delta = 0.95))
```

- formula syntax matches that of lme4
- form | fun(variable)
    - cens: handles censored data
    - weights and disp: allow different sorts of weighting
    - se: specify known standard errors primarily for meta-analysis
    - trunc： define truncation boundaries
    - trials： for binomial models
    - cat： specify the number of categories for ordinal models
- Group-level terms are of the form (coefs | group)

brms family

- Linear and robust linear regression: gaussian or student family with the identity link
- categorical - bernoulli, binomial, and categorical with logit link
- count data - poisson, negbinomial, and geometric
- survival regression - lognormal, Gamma, exponential, and weibull
- Ordinal regression - cumulative, cratio, sratio, and acat
- Excess zeros response - zero_inflated_poisson, zero_inflated_negbinomial, zero_inflated_binomial, zero_inflated_beta, hurdle_poisson, hurdle_negbinomial, and hurdle_gamma
- Specify custom families to include more distributions

brms parameters

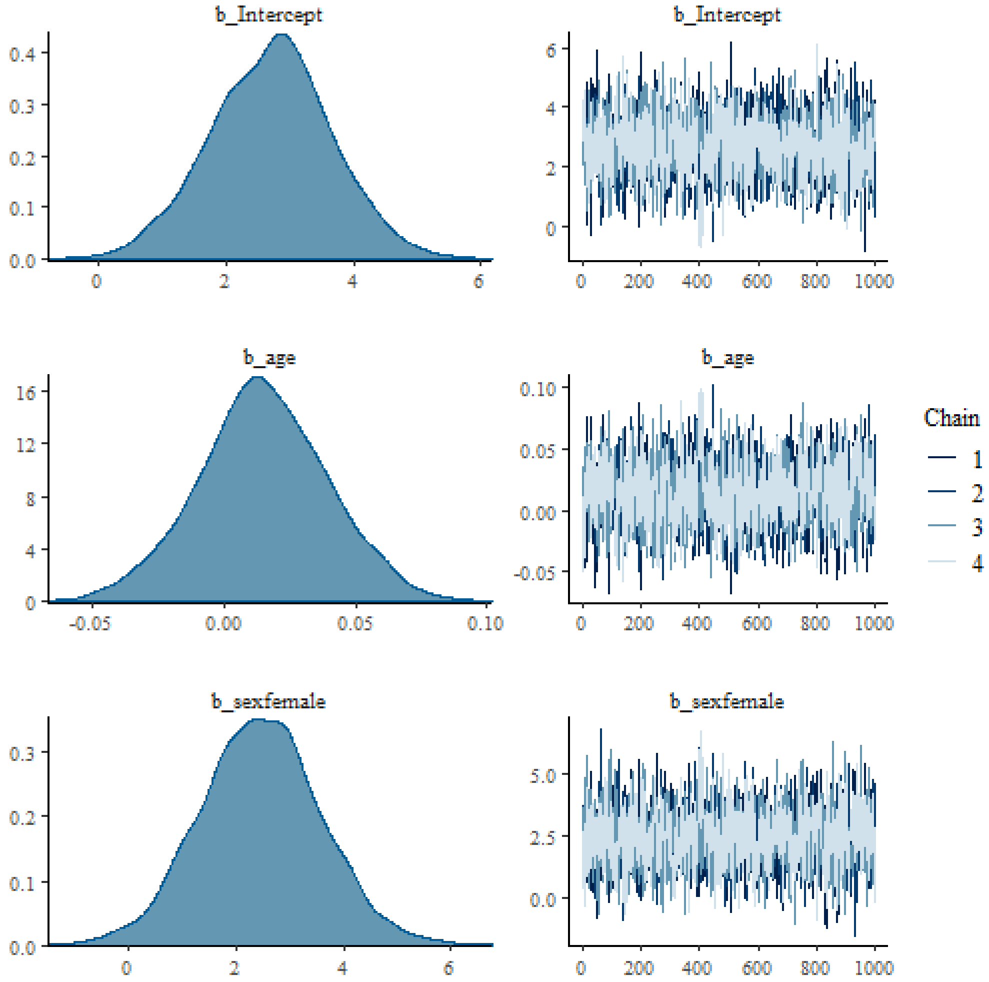
- prior: Prior distributions of model parameters

```
prior <- c(set_prior("normal(0,10)", class = "b", coef = "age"),
set_prior("cauchy(1,2)", class = "b", coef = "sexfemale"),
set_prior("student_t(3,0,5)", class = "sd", group="patient"))
```

- control: Adjusting the sampling behavior of Stan
  - choosing the number of iterations, warmup samples, and chains,
  - decrease (or eliminate at best) the number of divergent transitions that cause a bias in the obtained posterior samples. "There were x divergent transitions after warmup."

Check Posterior distribtion

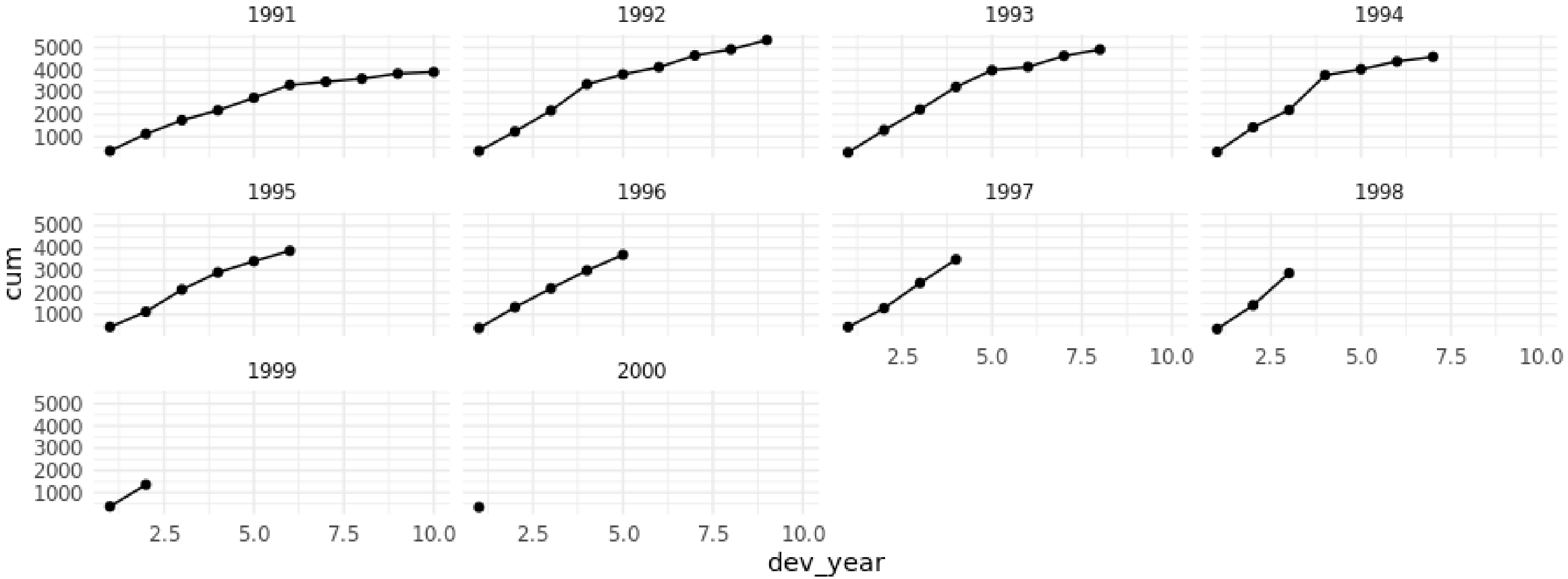
Posterior distribtion of parameters and chain convergence need to be checked



Example

non-linear model to model insurance loss

- Data



- Model

$$cum_{AY,dev} \sim N(\mu_{AY,dev}, \sigma)$$

$$\mu_{AY,dev} = ult_{AY}(1 - exp(-(\frac{dev}{\theta})^{w}))$$

Prior parameter distributions

- Prior parameter distribution

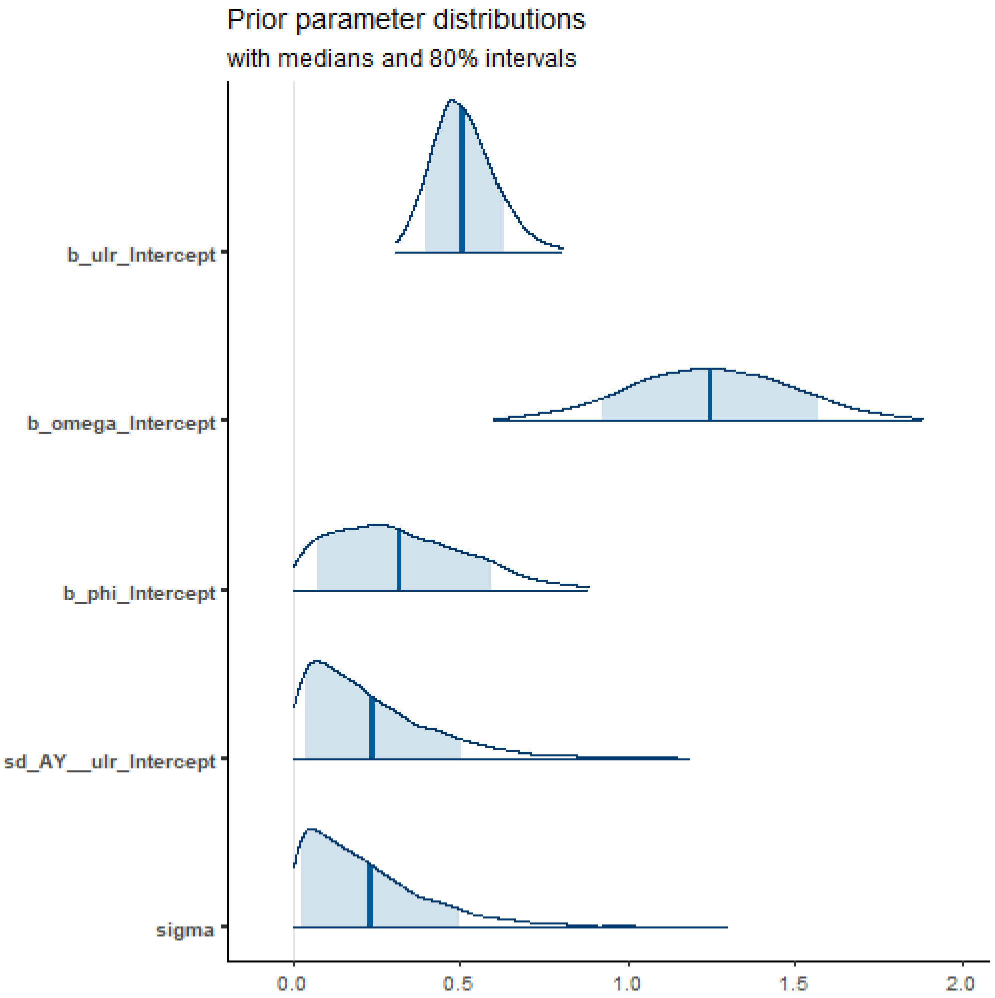
$$\phi \sim N(0.25, 0.25^2)^+$$

$$\omega \sim N(1.25, 0.25^2)^+$$

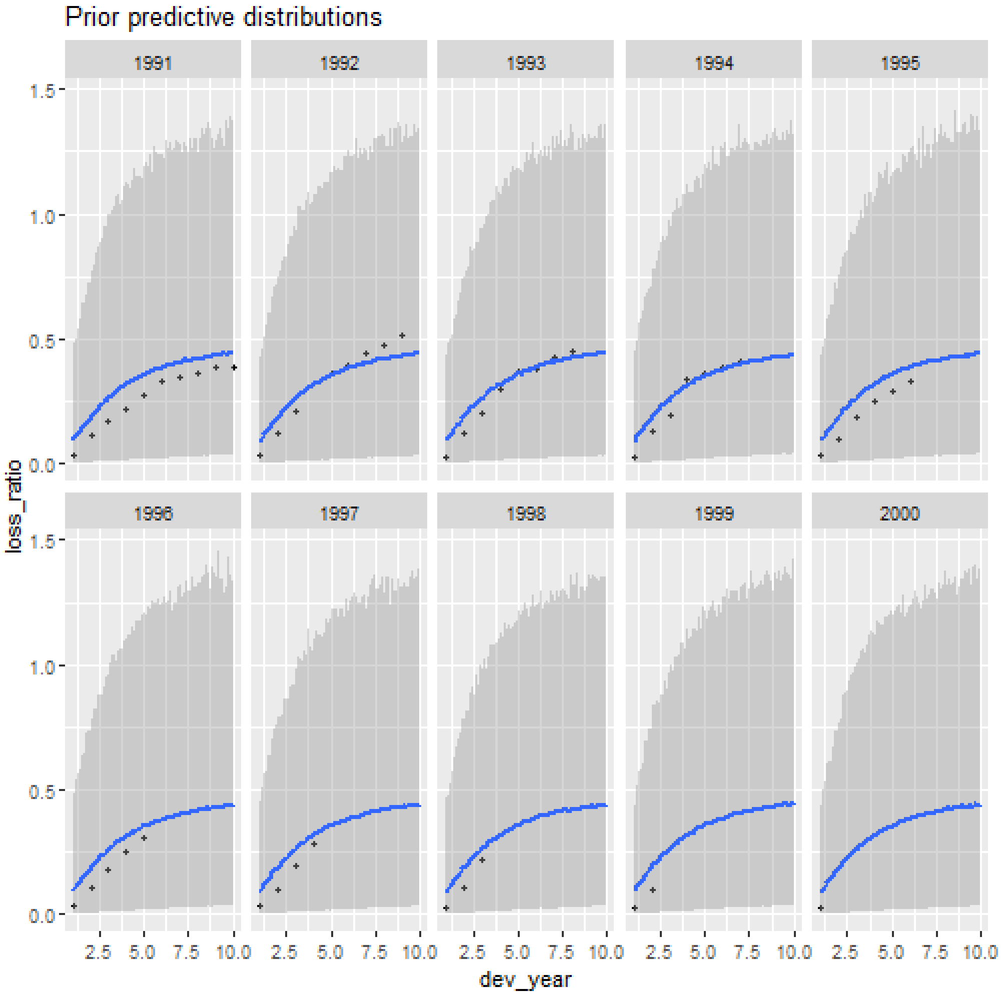
$$\gamma \sim logN(log(0.5), log(1.2)^2)$$

$$\sigma \sim Student - t(5, 0, 0.25)^+$$

$$\sigma_{\gamma^0} \sim Student - t(5, 0, 0.25)^+$$



Estimate on Prior predictive distribution

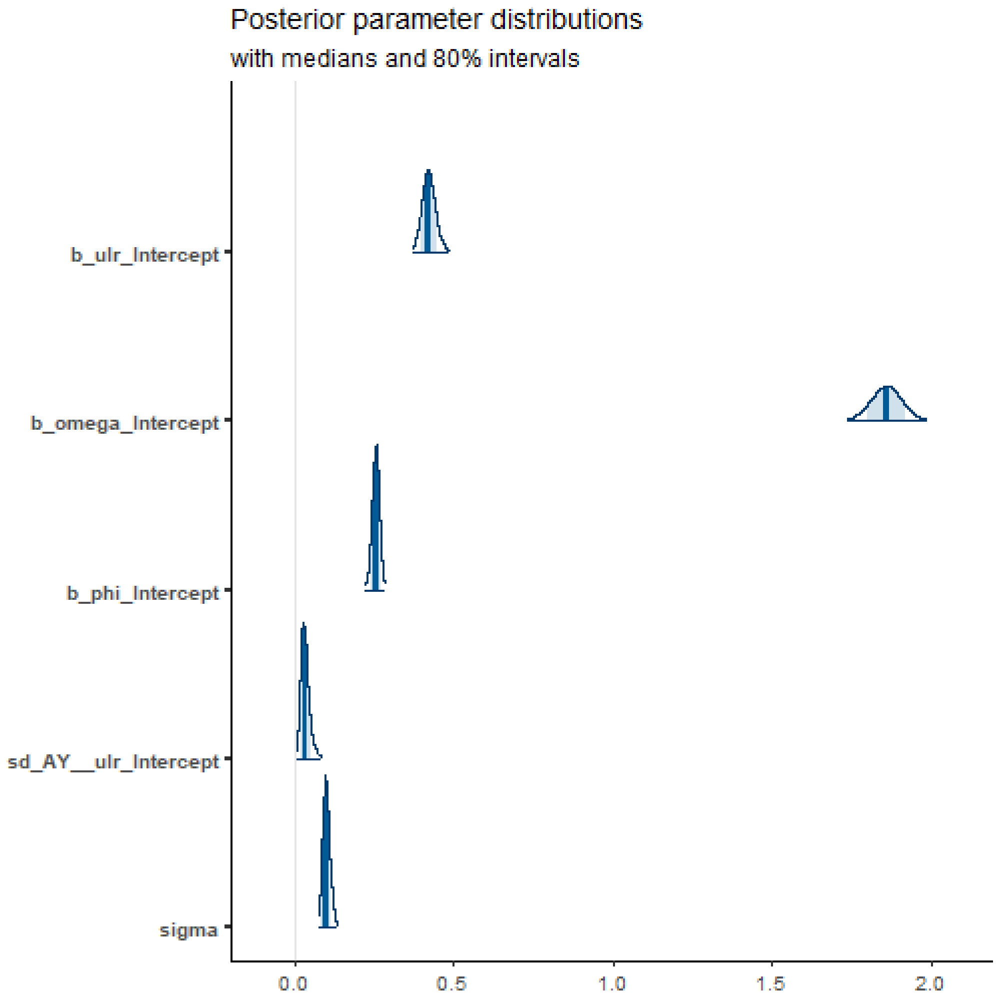


Model

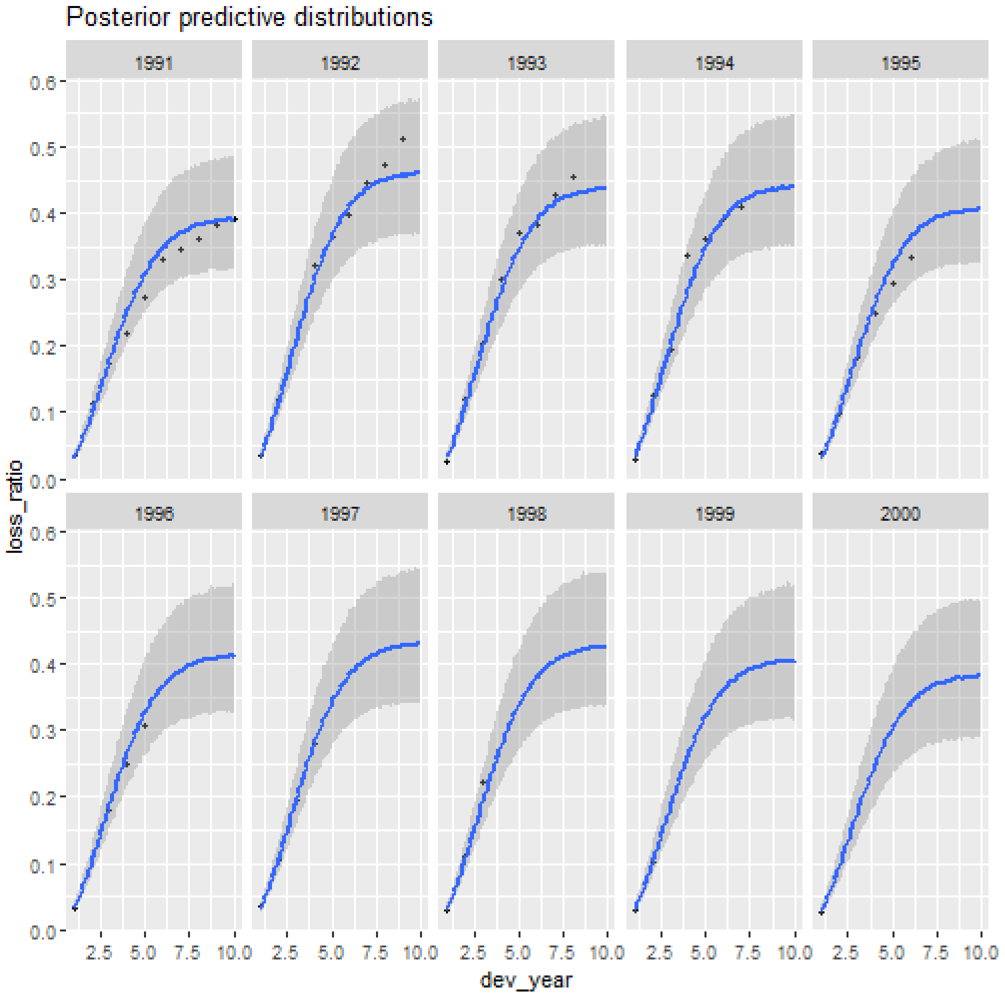
```
nlform2 <- bf(loss_ratio ~ log(ulr * (1 - exp(-(dev_year*phi)^omega))),
ulr ~ 1 + (1|AY), omega ~ 1, phi ~ 1,
nl = TRUE)

m2 <- brm(nlform2, data = loss,
family = lognormal(link = "identity", link_sigma = "log"),
prior = c(
prior(lognormal(log(0.5), log(1.2)), nlpar = "ulr", lb=0),
prior(normal(1.25, 0.25), nlpar = "omega", lb=0),
prior(normal(0.25, 0.25), nlpar = "phi", lb=0),
prior(student_t(5, 0, 0.25), class = "sigma"),
prior(student_t(5, 0, 0.25), class = "sd", nlpar="ulr")
),
sample_prior = "no", seed = 1234
)
```

Posterior parameter distributions



Posterior predictive distributions



Pros and Cons of Bayesian method

- Pros
    - more thoughtfull on data generating models/ process
    - Include prior knowledge
    - more flexible in model structure
- Cons
    - different mindset and could be difficult to set up
    - more time consuming to fit models
    - hard to scale to very large data

links

https://mc-stan.org/

https://cran.r-project.org/web/packages/brms/index.html