Programa Universitario de Estudios del Desarrollo
UNAM

Clase 11: Estimación en R-Stan

Dr. Héctor Nájera
Dr. Curtis Huffman

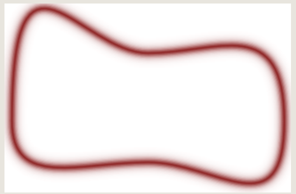
Pasos en inferencia bayesiana

Pregunta de investigación (Teoría)

Hipótesis β

Verosimilitud
Datos + Modelo que los explica
P(D|B)
$Y = a + \hat{\beta} X$

Espacio típico (Donde se muestrea la posterior)



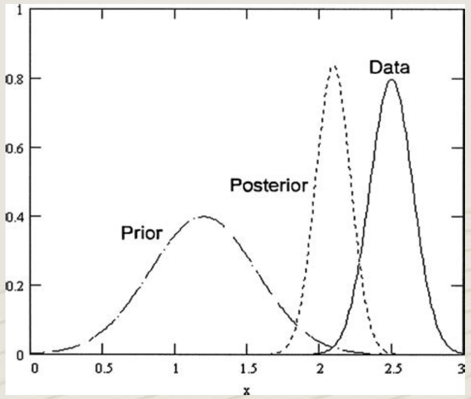
Priors sobre H
$\beta \sim N(0,10)$

Exploración con MCMC

Evaluación de las cadenas: Rhat y ESS

Distribución posterior de $\beta*$

Evaluación de H



Stan: Diferentes plataformas

RStan

The R interface to Stan

Stan está escrito en C++

Programas

- rstan
- PyStan
- Stan.jl
- CmdStan

rstan:

- Rstan (crudo)
- rstan (interfaces para rstan)

Interfaces para rstan:

- rstanarm
- rethinking
- brms
- cmdstanr

Complementos:

- Tidybayes
- bayesplot

Avanzado: mayor potencia y flexibilidad

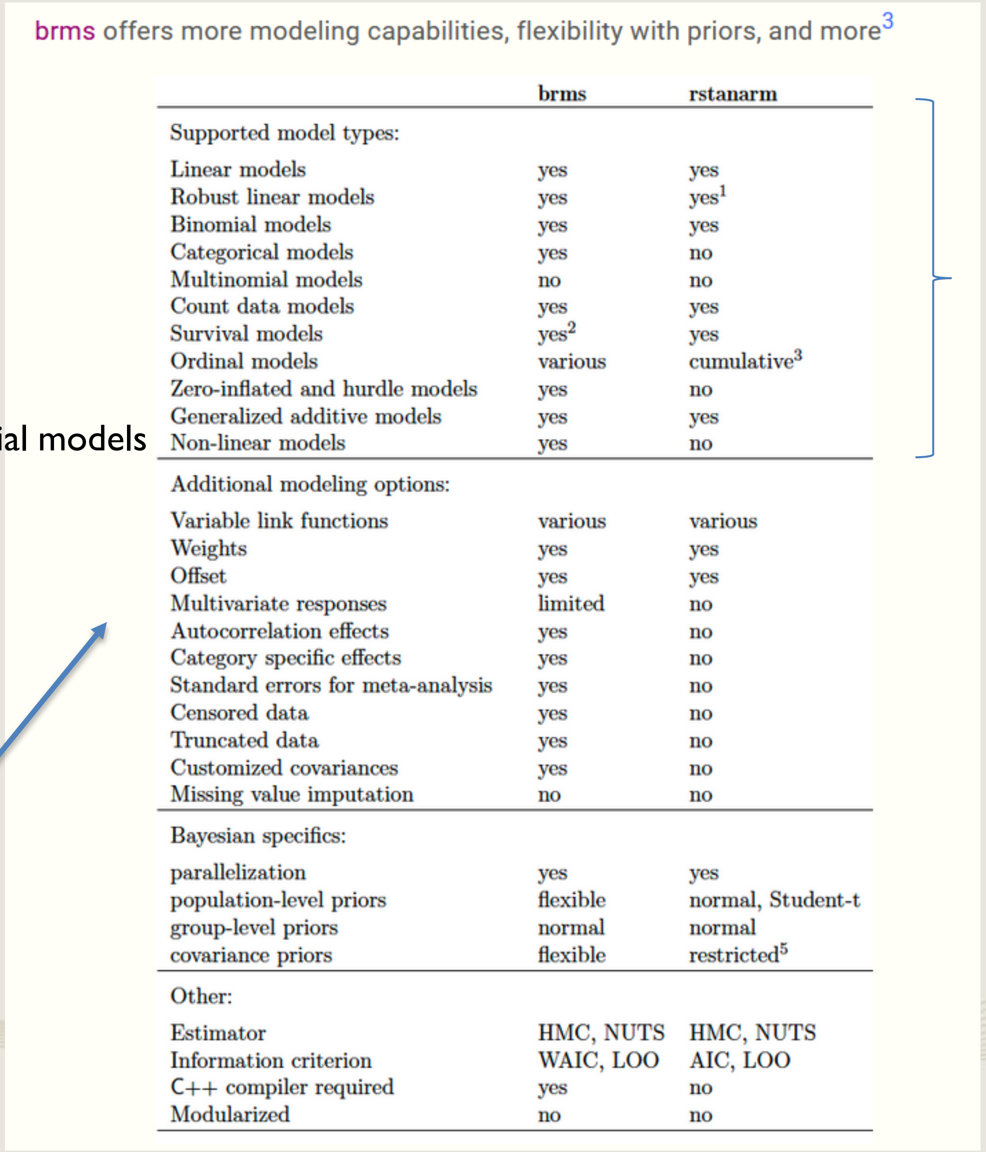
Intermedio: menor flexibilidad

Intermediarios: brms y rstanarm

Además brms conecta con cmdstanr y puede ser multicore y multithread –y ahora se puede usar con las tarjetas gráficas- dentro de cada cadena!

Noten las combinaciones posibles:
- Modelo espacio/temporal (AR y CAR) inflado de ceros!!!!

Spatial models


brms offers more modeling capabilities, flexibility with priors, and more[3]

	brms	rstanarm
Supported model types:		
Linear models	yes	yes
Robust linear models	yes	yes[1]
Binomial models	yes	yes
Categorical models	yes	no
Multinomial models	no	no
Count data models	yes	yes
Survival models	yes[2]	yes
Ordinal models	various	cumulative[3]
Zero-inflated and hurdle models	yes	no
Generalized additive models	yes	yes
Non-linear models	yes	no
Additional modeling options:		
Variable link functions	various	various
Weights	yes	yes
Offset	yes	yes
Multivariate responses	limited	no
Autocorrelation effects	yes	no
Category specific effects	yes	no
Standard errors for meta-analysis	yes	no
Censored data	yes	no
Truncated data	yes	no
Customized covariances	yes	no
Missing value imputation	no	no
Bayesian specifics:		
parallelization	yes	yes
population-level priors	flexible	normal, Student-t
group-level priors	normal	normal
covariance priors	flexible	restricted[5]
Other:		
Estimator	HMC, NUTS	HMC, NUTS
Information criterion	WAIC, LOO	AIC, LOO
C++ compiler required	yes	no
Modularized	no	no

Si no se puede hay que usar rstan

¿Intermediario?

- Brms:

```
M1.1<-brm(ins_ali_mo_3 ~ ed_jefe_2 + ed_jefe_3 +   ed_jefe_4 + ed_jefe_5
+ sexo_jefe_1 + edad_jefe + niv_ed_prom + tv + compu +
refri   +     lavad  +     tvpaga    +  hablaind  +  tinaco  + internet + tot_integ +
salario_mun +
i_nolav +     i_nostvp + i_noref + salario_ent + rural + (1|id_mun) +
(1|clave_ent),
data=D1, family = bernoulli(),
prior = c(set_prior("normal(0,1)", class = "b"),
set_prior("normal(0,5)", class= "sd")),
warmup = 400, iter = 1800, control = list(adapt_delta = .96),
chains = 2, cores=14, threads = threading(14, grainsize = 100),backend =
"cmdstanr")
```

STAN:

```
functions {
  /* integer sequence of values
   * Args:
   *   start: starting integer
   *   end: ending integer
   * Returns:
   *   an integer sequence from start to end
   */
  int[] sequence(int start, int end) {
    int seq[end - start + 1];
    for (n in 1:num_elements(seq)) {
      seq[n] = n + start - 1;
    }
    return seq;
  }
  // compute partial sums of the log-likelihood
  real partial_log_lik_lpmf(int[] seq, int start, int end, data int[] Y, data matrix Xc, vector b, real Intercept, data int[] J_1, data vector
Z_1_1, vector r_1_1, data int[] J_2, data vector Z_2_1, vector r_2_1) {
    real ptarget = 0;
    int N = end - start + 1;
    // initialize linear predictor term
    vector[N] mu = Intercept + rep_vector(0.0, N);
    for (n in 1:N) {
      // add more terms to the linear predictor
      int nn = n + start - 1;
      mu[n] += r_1_1[J_1[nn]] * Z_1_1[nn] + r_2_1[J_2[nn]] * Z_2_1[nn];
    }
    ptarget += bernoulli_logit_glm_lpmf(Y[start:end] | Xc[start:end], mu, b);
    return ptarget;
  }
}
data {
  int<lower=1> N;  // total number of observations
  int Y[N];  // response variable
  int<lower=1> K;  // number of population-level effects
  matrix[N, K] X;  // population-level design matrix
  int grainsize;  // grainsize for threading
  // data for group-level effects of ID 1
  int<lower=1> N_1;  // number of grouping levels
  int<lower=1> M_1;  // number of coefficients per level
  int<lower=1> J_1[N];  // grouping indicator per observation
```

STAN Cont.

```
STAN
 // group-level predictor values
  vector[N] Z_1_1;
  // data for group-level effects of ID 2
  int<lower=1> N_2;  // number of grouping levels
  int<lower=1> M_2;  // number of coefficients per level
  int<lower=1> J_2[N];  // grouping indicator per observation
  // group-level predictor values
  vector[N] Z_2_1;
  int prior_only;  // should the likelihood be ignored?
}
transformed data {
  int Kc = K - 1;
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  int seq[N] = sequence(1, N);
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // population-level effects
  real Intercept;  // temporary intercept for centered predictors
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  vector[N_1] z_1[M_1];  // standardized group-level effects
  vector<lower=0>[M_2] sd_2;  // group-level standard deviations
  vector[N_2] z_2[M_2];  // standardized group-level effects
}
transformed parameters {
  vector[N_1] r_1_1;  // actual group-level effects
  vector[N_2] r_2_1;  // actual group-level effects
  r_1_1 = (sd_1[1] * (z_1[1]));
  r_2_1 = (sd_2[1] * (z_2[1]));
}
model {
  // likelihood including constants
  if (!prior_only) {
    target += reduce_sum(partial_log_lik_lpmf, seq, grainsize, Y, Xc, b, Intercept, J_1, Z_1_1, r_1_1, J_2, Z_2_1, r_2_1);
  }
  // priors including constants
  target += normal_lpdf(b | 0,1);
  target += student_t_lpdf(Intercept | 3, 0, 2.5);
  target += normal_lpdf(sd_1 | 0,5)
```

Estructura de modelo bayesiano

Estructura de un modelo Bayesiano:

$$\begin{aligned}
\text{outcome}_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \beta \times \text{predictor}_i \\
\beta &\sim \text{Normal}(0, 10) \\
\sigma &\sim \text{HalfCauchy}(0, 1)
\end{aligned}$$

$$\begin{aligned}
D_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta_A A_i \\
\alpha &\sim \text{Normal}(10, 10) \\
\beta_A &\sim \text{Normal}(0, 1) \\
\sigma &\sim \text{Uniform}(0, 10)
\end{aligned}$$

$$\begin{aligned}
D_i &\sim \text{Normal}(\mu_i, \sigma) && \text{[likelihood]} \\
\mu_i &= \alpha + \beta_R R_i + \beta_A A_i && \text{[linear model]} \\
\alpha &\sim \text{Normal}(10, 10) && \text{[prior for } \alpha\text{]} \\
\beta_R &\sim \text{Normal}(0, 1) && \text{[prior for } \beta_R\text{]} \\
\beta_A &\sim \text{Normal}(0, 1) && \text{[prior for } \beta_A\text{]} \\
\sigma &\sim \text{Uniform}(0, 10) && \text{[prior for } \sigma\text{]}
\end{aligned}$$

https://mc-stan.org/docs/stan-users-guide/linear-regression.html

Pasos para la modelación

Especificación del modelo (likelihood y priors)

Gibbs:
- Rjags + Jags(windows)

Hamiltonian: Rstan + Interface de su preferencia

RStan

The R interface to Stan

1. Preparación de datos
2. Escritura del modelo (verosimilitud y prior)
3. Compilación
4. Estimación de MCMC
5. Inspección de resultados
6. Diagnósticos

Software

- Programa: R + Rstudio
- Algoritmo HMC: Rstan
- Interface: brms + cmdstan

CmdStanR

Paso 1

- Instalar (asegurarse) R version 4 o superior:

https://cran.r-project.org/bin/windows/base/

Paso 2

- Instalar Rstudio

https://www.rstudio.com/products/rstudio/download/#download

Paso 3

- Instalar Rtools 4.0 o superior

https://cran.r-project.org/bin/windows/Rtools/

Paso 4

- Instalar Rstan

Seguir estos pasos

https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started

Asegurarse que el ejemplo "corre"

Paso 5

- Instalar en Rstudio

Brms, tidybayes, bayesplot, posterior, tidyverse

Paso 6

- Instalar cmdstanr –mayor rapidez-

https://mc-stan.org/cmdstanr/

Introducción –refrescando- R Software

- Algunas guías
  - https://www.youtube.com/watch?v=_V8eKsto3Ug