An amateur's adventures in BayesiStan

Or: trancending binary oppositions like a pro

Torkild H. L.
2019-03-15

Contents

What is Stan, and why would you use it?

Amateur answer: A program that let you estimate models "the bayesian way",

- which means the results are represented as the posterior distribution of parameters,
- which means your results are a data set about the parameters.

Pro answer: A fairly new, updated MCMC sampler that uses Hamiltonian Monte Carlo.

- The future of bayesian inference?

My research problem

Has family background lost (some of) its importance for education over birth cohorts?

Are innate abilities/personality/etc more important now than they were, say, 50 yrs ago?

Context: Nordic countries open, education free of charge/accessible

Status quo in sociology: Same old effects of social background.

But, often not with control for genetics!

I want to answer look at these issues across cohorts born in the 20th Century

Data from the Norwegian twin panel (b.1915-1991).

Why am I interested in Stan?

Bayesian inference is appealing

Trancend binary opposition of significant vs. non-significant

Real credibility intervals for uncertainty!

Can ask posterior distribution questions: How likely is it that genes >> family in importance?

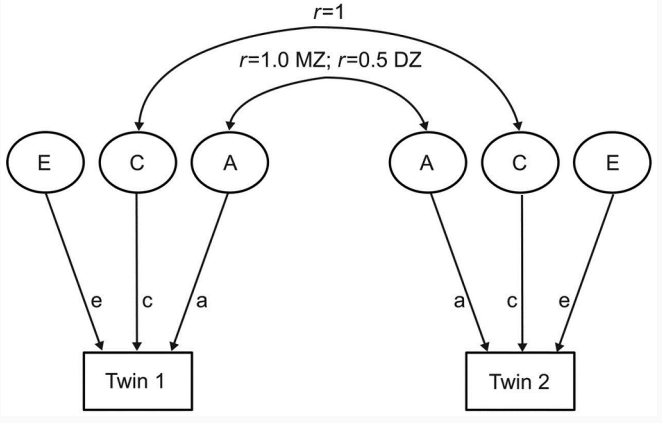
Allows specifying a model with "meta-parameters" (tranformed parameters)

Prior distributions on parameters

Easy to regularize: Avoid negative variances and impossible predictions.

My use case: Twin models of education

- I want to estimate "twin models" using data on Mz and Dz twins.



- I am interested in A, C and E: The *variance components* that sums to 100%
- Typically specified as structural equation models with latent variables (in e.g. OpenMX)
- But can also be specified as a mixed model.

The Classic Twin Design is also a mixed

- Mixed model == multilevel model == HLM == random effects model
- Simple data structure:
  - Really only three variables: Y for twin1, Y for twin2 and their zygosity
  - But twins nested in *j* families, of Dz or Mz types
  - Our Y is educational attainment in years
- A mixed model formulation of CTD:

For MZ twins, as before, we have

$$y_{ij} = \mu + \alpha_j + c_j + e_{ij}$$

For DZ twins, we now write:

$$y_{ij} = \mu + \sqrt{0.5}\alpha_j + c_j + \epsilon_{ij}$$

where $\epsilon_{ij} = e_{ij} + \sqrt{0.5}\alpha_{ij}$

Since $Var(\epsilon_{ij}) = Var(e_{ij}) + 0.5Var(\alpha_{ij})$, this new noise term will have a standard deviation of $\sqrt{\sigma_e^2 + 0.5\sigma_A^2}$.

The data

```
### Read in simulated data on 2000 twinpairs
simtwins <- fread(input = here("simtwins.csv"))
head(simtwins)
```

```
##            tw1        tw2 mz dz fam
## 1: -1.3433212 -0.6607416  1  0   1
## 2: -0.6528377 -0.6607416  0  1   2
## 3:  1.4186128  0.7214556  0  1   3
## 4:  1.4186128 -0.6607416  0  1   4
## 5:  0.7281293  0.7214556  1  0   5
## 6:  0.7281293 -0.6607416  0  1   6
```

Data prep for Stan

```
data_stan = list(n_fam = max(simtwins$fam),
                 n_famtw_mz = sum(simtwins$mz == 1),
                 n_famtw_dz = sum(simtwins$mz == 0),
                 y_mz = c(simtwins[mz == 1,]$tw1,
                          simtwins[mz == 1,]$tw2),
                 y_dz = c(simtwins[mz == 0,]$tw1,
                          simtwins[mz == 0,]$tw2),
                 fam_mz = c(simtwins[mz == 1,]$fam,
                            simtwins[mz == 1,]$fam),
                 fam_dz = c(simtwins[mz == 0,]$fam,
                            simtwins[mz == 0,]$fam),
                 outcome_sd = sd(c(simtwins$tw1, simtwins$tw2)),
                 outcome_mean = mean(c(simtwins$tw1, simtwins$tw2)))
```

Stan expects data as a list. Contents of the list is defined in the Stan model definition.

The Stan call in R

```
dirichlet_fit <- stan(file = here::here("./mixed-ace-dirichlet.stan"),
           data = data_stan, iter=10000, pars = interesting)
save(dirichlet_fit, file="draws20000.RData")
```

- `iter`: 10000 iterations from the posterior (but half reserved for "warmup" phase)
- model saves values from the `pars` list of parameters. It's mu, a, c, e_sigma, A, C, E, Asd, Csd, Esd
- The model is specified in the `stan` file: `mixed-ace-dirichlet.stan`

The Stan model, data part

```
data {
  int <lower=0> n_fam; // number of families
  int <lower=0> n_famtw_mz; // number of mz nested twins in families
  int <lower=0> n_famtw_dz; // number of dz nested twins in families
  real y_mz[n_famtw_mz * 2]; // responses
  real y_dz[n_famtw_dz * 2]; // responses
  int fam_mz[n_famtw_mz * 2]; // family indicator (1:nfam)
  int fam_dz[n_famtw_dz * 2]; // family indicator (1:nfam)
  real<lower = 0> outcome_sd;
  real outcome_mean;
}
transformed data {
  vector[3] dirichlet_prior;
  dirichlet_prior = rep_vector(1, 3);
}
```

Parameters to be estimated

```
parameters {
  // mean
  real mu;
  // overall variance
  real<lower = 0> sigma;
  // variance components
  simplex[3] var_comp_shares;
  // "random-effects" sd for genetics
  vector[n_fam] a_shared_std;
  // random effects sd for common env
  vector[n_fam] c_shared_std;
}
```

...and transformed parameters

```
transformed parameters {
  // "random effects"
  vector[n_fam] a_shared;
  vector[n_fam] c_shared;
  real A;
  real C;
  real E;
  real Asd;
  real Csd;
  real Esd;
  real a;
  real c;
  real e_sigma;
```

and more trans. parameters

```
  Asd = var_comp_shares[1];
  Csd = var_comp_shares[2];
  Esd = var_comp_shares[3];
  A = Asd * sigma^2;
  C = Csd * sigma^2;
  E = Esd * sigma^2;
  a = sqrt(A);
  c = sqrt(C);
  e_sigma = sqrt(E);
  a_shared = a * a_shared_std;
  c_shared = c * c_shared_std;
}
```

The actual model

```
model {
  vector[n_famtw_mz * 2] y_mz_expected;    // declare space for expected values of y
  vector[n_famtw_dz * 2] y_dz_expected;
  mu ~ normal(outcome_mean, outcome_mean * 0.2);    // prior for mu
  sigma ~ normal(outcome_sd, outcome_sd * 0.3);     // prior for total phenotypic variation
  var_comp_shares ~ dirichlet(dirichlet_prior);     // the variance components sums to 1
  a_shared_std ~ normal(0,1);
  c_shared_std ~ normal(0,1);
  // model
  for (i in 1:(n_famtw_mz * 2)){
    y_mz_expected[i] = mu + a_shared[fam_mz[i]] + c_shared[fam_mz[i]];
  }
  for (i in 1:(n_famtw_dz * 2)){
    y_dz_expected[i] = mu + sqrt(0.5) * a_shared[fam_dz[i]] + c_shared[fam_dz[i]];
  }
  target += normal_lpdf(y_mz | y_mz_expected, e_sigma);
  target += normal_lpdf(y_dz | y_dz_expected, sqrt(E + 0.5 * A));
}
```

Output

```
print(dirichlet_fit)
```

```
## Inference for Stan model: mixed-ace-dirichlet.
## 4 chains, each with iter=20000; warmup=10000; thin=1;
## post-warmup draws per chain=10000, total post-warmup draws=40000.
##
##             mean se_mean    sd     2.5%      25%      50%      75%
## mu          0.06    0.00  0.01     0.04     0.05     0.06     0.06
## a           0.64    0.00  0.04     0.55     0.61     0.64     0.67
## c           0.45    0.00  0.05     0.33     0.42     0.45     0.49
## sigma       0.98    0.00  0.01     0.96     0.97     0.98     0.99
## e_sigma     0.59    0.00  0.01     0.56     0.58     0.59     0.60
## A           0.41    0.00  0.06     0.31     0.38     0.41     0.45
## C           0.20    0.00  0.05     0.11     0.17     0.20     0.24
## E           0.35    0.00  0.02     0.32     0.34     0.35     0.36
## Asd         0.43    0.00  0.06     0.32     0.39     0.43     0.47
## Csd         0.21    0.00  0.05     0.12     0.18     0.21     0.24
## Esd         0.36    0.00  0.02     0.33     0.35     0.36     0.37
## lp__    -6129.87    0.91 63.14 -6254.92 -6171.78 -6129.41 -6087.61
##            97.5% n_eff Rhat
## mu          0.07 59585    1
```

How to assess convergence?

We want to:

- Base conclusions on posterior distributions
- Report accurate estimates and uncertainties
- Avoid writing erratums

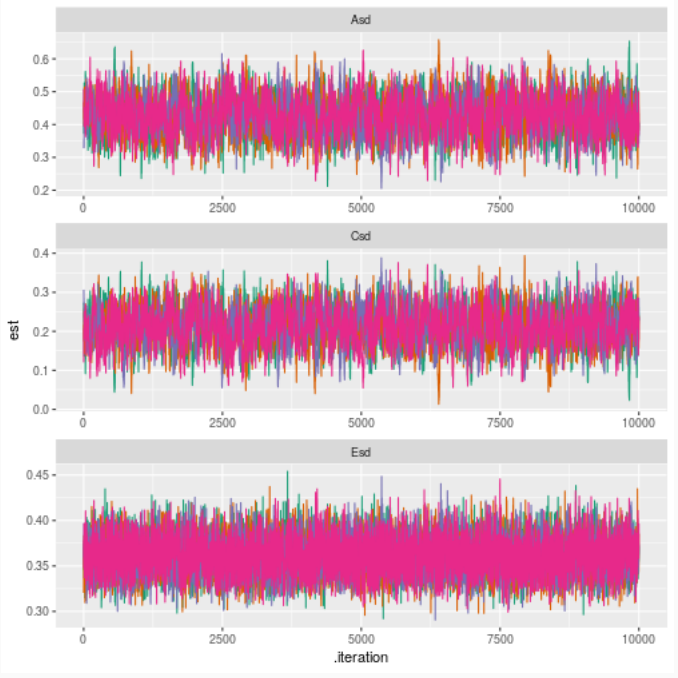
If a MCMC chain converges, we're good.

How do we know it converged?

Practical diagnostics

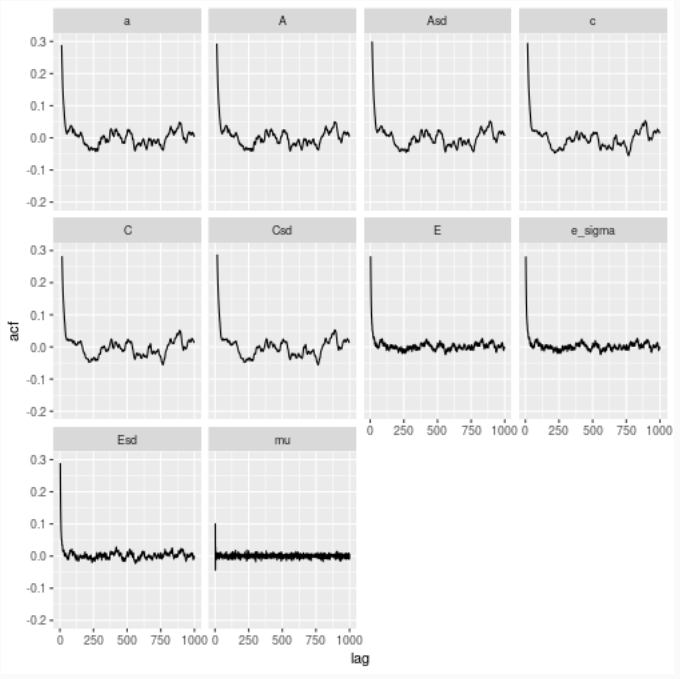
- RStan has built-in functions, e.g. `check_hmc_diagnostics()`
- Compare results by chains
- Look at traceplots (plots of MCMC chains)
- Calc. stats, e.g. R-hat
- Examine autocorrelation

Traceplots



Autocorrelation functions

```
## Warning: Removed 12 rows containing missing values (geom_path).
```



Posteriors for VCs

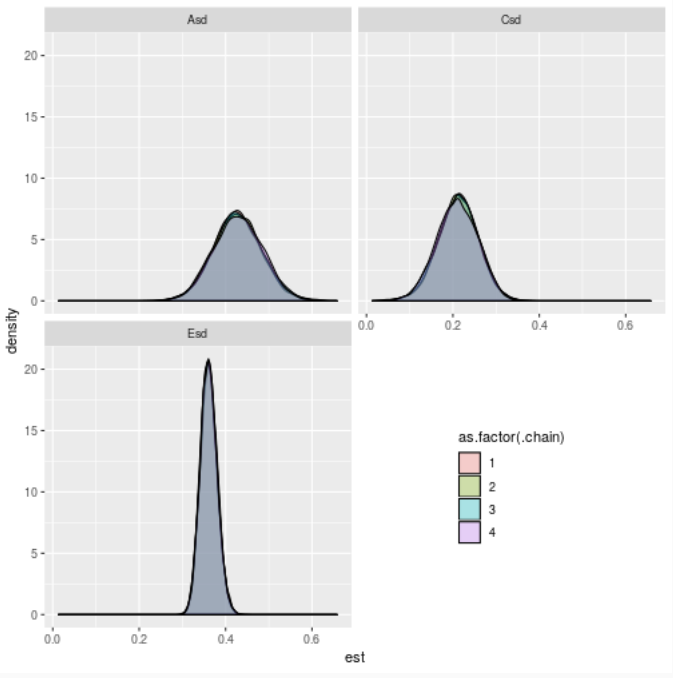
```
posterior <- tidy_draws(dirichlet_fit)
head(select(posterior, interesting))
```

```
## # A tibble: 6 x 10
##       mu     a     c e_sigma     A     C     E   Asd   Csd   Esd
##    <dbl> <dbl> <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 0.0405 0.621 0.481   0.603 0.386 0.231 0.364 0.394 0.236 0.371
## 2 0.0719 0.631 0.490   0.581 0.398 0.240 0.337 0.408 0.246 0.346
## 3 0.0685 0.604 0.516   0.589 0.365 0.266 0.347 0.373 0.272 0.355
## 4 0.0456 0.632 0.486   0.599 0.399 0.236 0.359 0.401 0.237 0.361
## 5 0.0566 0.640 0.444   0.602 0.410 0.197 0.362 0.423 0.203 0.374
## 6 0.0562 0.651 0.417   0.590 0.424 0.174 0.348 0.448 0.184 0.368
```

```
histoace <- posterior %>%
  select(Asd, Csd, Esd, .chain) %>%
  gather(key=parm, value=est, -.chain) %>%
  ggplot() + geom_density(aes(fill=as.factor(.chain), x=est), alpha=.3) + facet_wrap(~parm,
```

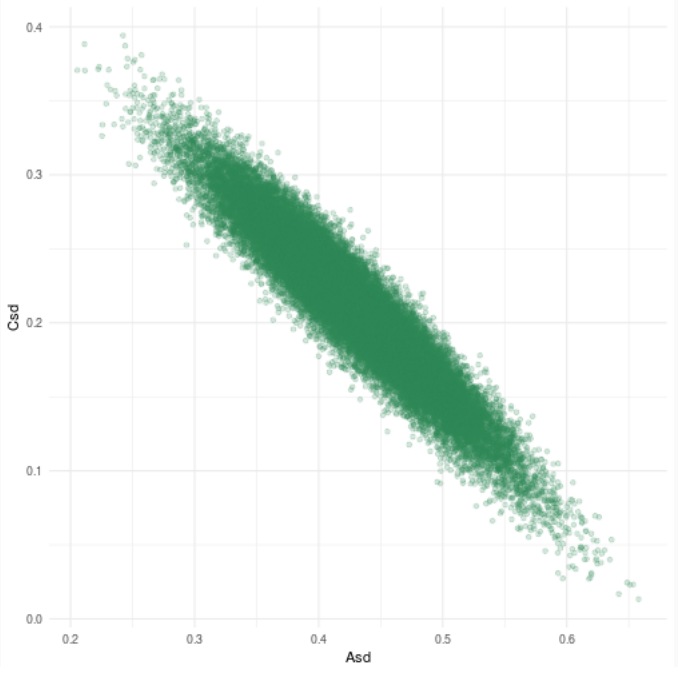
Posteriors for VCs

```
shift_legend2(histoace)
```



Scatter of posterior of Asd and Csd

```
ggplot(posterior) + geom_point(aes(x=Asd, y=Csd), alpha=0.2, fill= "seagreen", color="seagr
```



Ask the posterior questions!

```
# How likely that genetics more important than families?
AoverC <- nrow(posterior %>% filter(Asd>Csd))/nrow(posterior)
scales::percent(AoverC)
```

```
## [1] "98.5%"
```

```
# How likely that families more important than other environmental factors?
CoverE <- nrow(posterior %>% filter(Csd>Esd))/nrow(posterior)
scales::percent(CoverE)
```

```
## [1] "0.0350%"
```

```
# How likely that genetics explain more than half of variance?
Ahalf <-  nrow(posterior %>% filter(Asd>0.5))/nrow(posterior)
scales::percent(Ahalf)
```

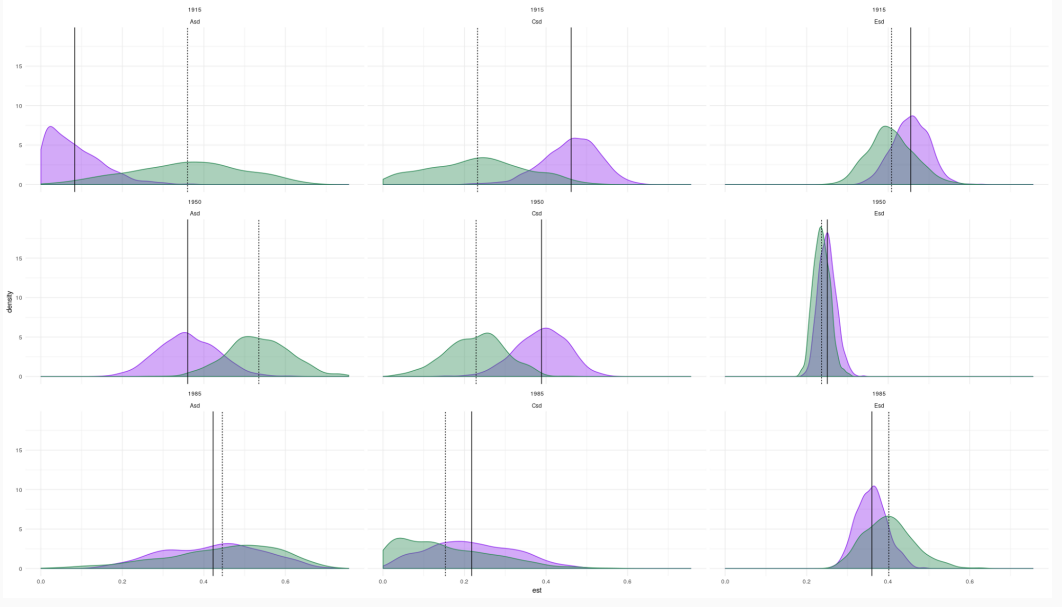
```
## [1] "10.1%"
```

Next steps

- Do more diagnostics
- Experiment more with priors
- Examine changes across 1915-1990 birth cohorts
- Extend model to
    - accept previous cohort's results as prior distribution?
    - incorporate changes over time in parameters??

Sneak peek @ 1915-1985 cohorts

Top to bottom: 1915,1950 and 1985 cohorts Left to right: A (genetics), C (family) and E (random) components Purple = women, green = men



- GitHub repo for Stan code: http://github.com/torkildl/bACEian