Lab 2

Alex Ziyu Jiang

Bayesian Binomial Models

In lecture we looked at a Bayesian binomial model on a dataset about labor market data. Today we look at its use on a different dataset. First, we download the package for the dataset:

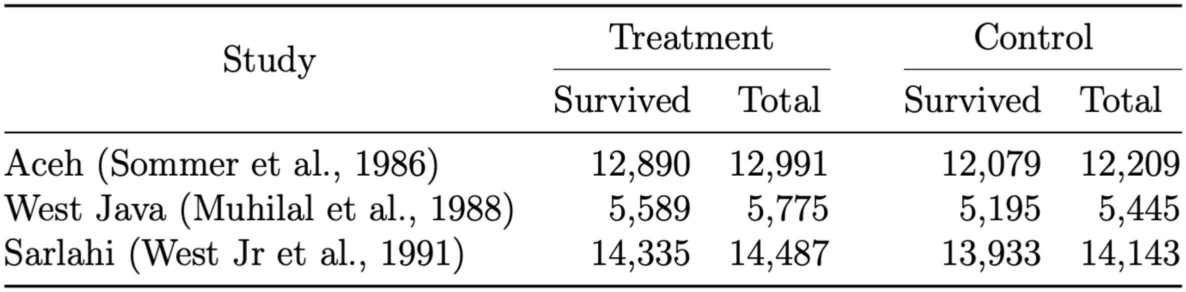
```
# install.packages("devtools") # run it for the first time
devtools::install_github("carloscinelli/generalizing")
```

```
## Skipping install of 'generalizing' from a github remote, the SHA1 (3c83d257) has not changed since la
##   Use `force = TRUE` to force installation
```

The dataset

Here we will be looking into three experiments that were designed to study the effects of vitamin A supplementation on childhood mortality. Three trials were carried outt in Aceh, West Java and Sarlahi. Based on scientific background, Vitamin A reduces childhood mortality by reducing the incidence, severity or duration of life-threatening diseases such as measles and diarrhea. We focus on the following measures:

- The outcome of the trial: Survival and death during the trial.
- The treatments: Treatment group if assigned vitamin A; control if assigned placebo.



Study	Treatment		Control	
	Survived	Total	Survived	Total
Aceh (Sommer et al., 1986)	12,890	12,991	12,079	12,209
West Java (Muhilal et al., 1988)	5,589	5,775	5,195	5,445
Sarlahi (West Jr et al., 1991)	14,335	14,487	13,933	14,143

Table 1: Observed data for the vitamin A studies.

The model

We are interested to study the contrast of survival rates between treatment and control groups and we can build a Bayesian model to study the differences. We denote P_1 as the survival rate in the treatment group, we further assume that the death of each subject in the treatment group is independent of each other and identical with probability $1 - P_1$ (which is the mortality rate). Suppose the total number of participants in the treatment group is N_1, the number of participants survived is n_1 (for the Aceh case, it will be $N_1 = 12,991, n_1 = 12,890$). Thus we can build a **binomial model** to characterize the number of participants surviving, in the treatment group:

$$n_1 \sim \text{Binom}(N_1, P_1)$$

We can do the similar thing for the control group and have the following model:

$$\begin{aligned} n_1 &\sim \text{Binom}(N_1, P_1) \\ n_0 &\sim \text{Binom}(N_0, P_0) \end{aligned}$$

Additionally, we should notice that P_0, P_1 are unknown values so we need to place priors on it in order to make it a Bayesian model. Keep in mind that they are essentially probabilities between zero and one, so it would make sense to give them a uniform prior on $[0, 1]$, without extra prior information.

The inference goal

There are a couple of quantities of interest that we would want to conduct inference on from this model. Obviously we want to look at the posterior distribution of P_0, P_1, the survival rates for the treatment and the control group, and we would also want to look at the **risk difference** $RD = (1 - P_1) - (1 - P_0) = P_0 - P_1$, the **risk ratio** $RR = \frac{1-P_1}{1-P_0}$, the **survival ratio** $1 - \frac{1-P_1}{1-P_0}$. The survival ratio can be interpreted as the percentage of children that were saved due to vitamin A supplementation.

The Jags implementation

```
library(rjags)
library(rethinking)
library(generalizing)
library(rstan)
# install.packages("devtools")
data("Aceh")
# JAGS code
model_code <- "
  model{
    # likelihood
    n1 ~ dbinom(P1, N1)
    n0 ~ dbinom(P0, N0)
    # prior
    P1 ~ dunif(0, 1)
    P0 ~ dunif(0, 1)
    # transformed quantities
    rd <- P0 - P1
    rr <- (1-P1)/(1-P0)
    sr <- (P1-P0)/(1-P0)
  }
"
# compile jags model
model <- jags.model(file = textConnection(model_code),
                    data = Aceh)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
```

```
##    Observed stochastic nodes: 2
##    Unobserved stochastic nodes: 2
##    Total graph size: 14
##
## Initializing model
```

```
samples <- coda.samples(model,
                        variable.names = c("P1", "P0", "rd", "rr", "sr"),
                        n.iter = 1e4)
samples <- as.data.frame(samples[[1]])
precis(samples, digits = 10)
```

```
##             mean           sd        5.5%         94.5%      histogram
## P0  0.989277260 0.0009275478  0.98778031  0.9907390296
## P1  0.992144532 0.0007794674  0.99087165  0.9933346700
## rd -0.002867271 0.0012051009 -0.00478789 -0.0009405565
## rr  0.738083157 0.0974209804  0.59433669  0.9031995249
## sr  0.261916843 0.0974209804  0.09680048  0.4056633062
```

Results

We see that the posterior distribution for both P_0, P_1 are centered pretty close to 1, and we have a risk difference posterior that is centered very close to zero, and a risk ratio posterior that is centered around 1. It seemed like the treatment effect is very small, but this is probably due to the fact the baseline survival rate is already very high. However, the survival ratio gives us more information, with its posterior mean around 0.26. This gives us the information that treatment of vitamin A prevented about 26% of deaths that would have otherwise occurred.

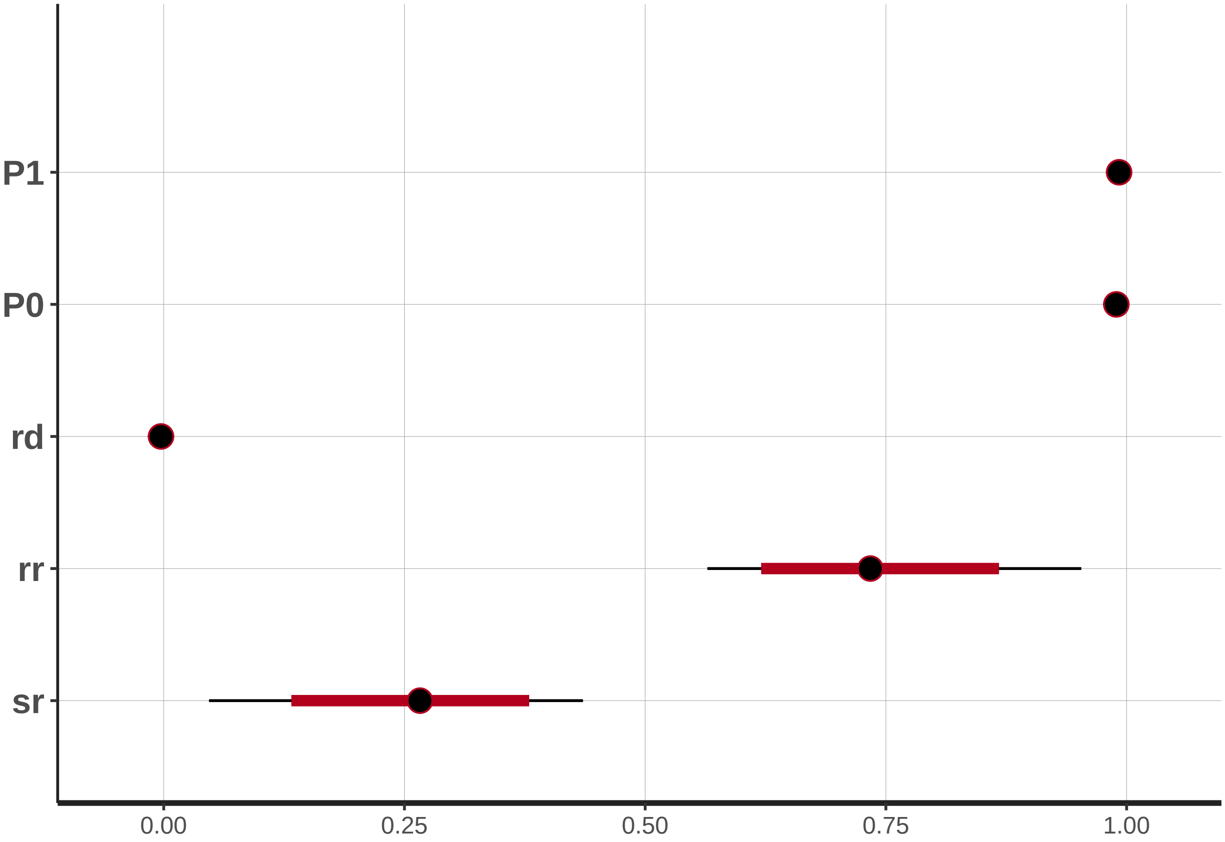
STAN code

Similarly, we can repeat the analysis in STAN. We can see from the results that they are very similar to each other.

```
## Running MCMC with 1 chain...
##
## Chain 1 Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 10000 [  1%]  (Warmup)
## Chain 1 Iteration:  200 / 10000 [  2%]  (Warmup)
## Chain 1 Iteration:  300 / 10000 [  3%]  (Warmup)
## Chain 1 Iteration:  400 / 10000 [  4%]  (Warmup)
## Chain 1 Iteration:  500 / 10000 [  5%]  (Warmup)
## Chain 1 Iteration:  600 / 10000 [  6%]  (Warmup)
## Chain 1 Iteration:  700 / 10000 [  7%]  (Warmup)
## Chain 1 Iteration:  800 / 10000 [  8%]  (Warmup)
## Chain 1 Iteration:  900 / 10000 [  9%]  (Warmup)
## Chain 1 Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1 Iteration: 1100 / 10000 [ 11%]  (Warmup)
## Chain 1 Iteration: 1200 / 10000 [ 12%]  (Warmup)
## Chain 1 Iteration: 1300 / 10000 [ 13%]  (Warmup)
## Chain 1 Iteration: 1400 / 10000 [ 14%]  (Warmup)
## Chain 1 Iteration: 1500 / 10000 [ 15%]  (Warmup)
## Chain 1 Iteration: 1600 / 10000 [ 16%]  (Warmup)
## Chain 1 Iteration: 1700 / 10000 [ 17%]  (Warmup)
## Chain 1 Iteration: 1800 / 10000 [ 18%]  (Warmup)
## Chain 1 Iteration: 1900 / 10000 [ 19%]  (Warmup)
## Chain 1 Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1 Iteration: 2100 / 10000 [ 21%]  (Warmup)
## Chain 1 Iteration: 2200 / 10000 [ 22%]  (Warmup)
## Chain 1 Iteration: 2300 / 10000 [ 23%]  (Warmup)
## Chain 1 Iteration: 2400 / 10000 [ 24%]  (Warmup)
## Chain 1 Iteration: 2500 / 10000 [ 25%]  (Warmup)
## Chain 1 Iteration: 2600 / 10000 [ 26%]  (Warmup)
## Chain 1 Iteration: 2700 / 10000 [ 27%]  (Warmup)
## Chain 1 Iteration: 2800 / 10000 [ 28%]  (Warmup)
## Chain 1 Iteration: 2900 / 10000 [ 29%]  (Warmup)
## Chain 1 Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1 Iteration: 3100 / 10000 [ 31%]  (Warmup)
## Chain 1 Iteration: 3200 / 10000 [ 32%]  (Warmup)
## Chain 1 Iteration: 3300 / 10000 [ 33%]  (Warmup)
## Chain 1 Iteration: 3400 / 10000 [ 34%]  (Warmup)
## Chain 1 Iteration: 3500 / 10000 [ 35%]  (Warmup)
## Chain 1 Iteration: 3600 / 10000 [ 36%]  (Warmup)
## Chain 1 Iteration: 3700 / 10000 [ 37%]  (Warmup)
## Chain 1 Iteration: 3800 / 10000 [ 38%]  (Warmup)
## Chain 1 Iteration: 3900 / 10000 [ 39%]  (Warmup)
## Chain 1 Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1 Iteration: 4100 / 10000 [ 41%]  (Warmup)
## Chain 1 Iteration: 4200 / 10000 [ 42%]  (Warmup)
## Chain 1 Iteration: 4300 / 10000 [ 43%]  (Warmup)
## Chain 1 Iteration: 4400 / 10000 [ 44%]  (Warmup)
## Chain 1 Iteration: 4500 / 10000 [ 45%]  (Warmup)
## Chain 1 Iteration: 4600 / 10000 [ 46%]  (Warmup)
## Chain 1 Iteration: 4700 / 10000 [ 47%]  (Warmup)
## Chain 1 Iteration: 4800 / 10000 [ 48%]  (Warmup)
## Chain 1 Iteration: 4900 / 10000 [ 49%]  (Warmup)
## Chain 1 Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1 Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1 Iteration: 5100 / 10000 [ 51%]  (Sampling)
## Chain 1 Iteration: 5200 / 10000 [ 52%]  (Sampling)
## Chain 1 Iteration: 5300 / 10000 [ 53%]  (Sampling)
## Chain 1 Iteration: 5400 / 10000 [ 54%]  (Sampling)
## Chain 1 Iteration: 5500 / 10000 [ 55%]  (Sampling)
## Chain 1 Iteration: 5600 / 10000 [ 56%]  (Sampling)
## Chain 1 Iteration: 5700 / 10000 [ 57%]  (Sampling)
## Chain 1 Iteration: 5800 / 10000 [ 58%]  (Sampling)
## Chain 1 Iteration: 5900 / 10000 [ 59%]  (Sampling)
## Chain 1 Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1 Iteration: 6100 / 10000 [ 61%]  (Sampling)
## Chain 1 Iteration: 6200 / 10000 [ 62%]  (Sampling)
## Chain 1 Iteration: 6300 / 10000 [ 63%]  (Sampling)
## Chain 1 Iteration: 6400 / 10000 [ 64%]  (Sampling)
## Chain 1 Iteration: 6500 / 10000 [ 65%]  (Sampling)
## Chain 1 Iteration: 6600 / 10000 [ 66%]  (Sampling)
## Chain 1 Iteration: 6700 / 10000 [ 67%]  (Sampling)
## Chain 1 Iteration: 6800 / 10000 [ 68%]  (Sampling)
## Chain 1 Iteration: 6900 / 10000 [ 69%]  (Sampling)
## Chain 1 Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1 Iteration: 7100 / 10000 [ 71%]  (Sampling)
## Chain 1 Iteration: 7200 / 10000 [ 72%]  (Sampling)
## Chain 1 Iteration: 7300 / 10000 [ 73%]  (Sampling)
## Chain 1 Iteration: 7400 / 10000 [ 74%]  (Sampling)
## Chain 1 Iteration: 7500 / 10000 [ 75%]  (Sampling)
## Chain 1 Iteration: 7600 / 10000 [ 76%]  (Sampling)
## Chain 1 Iteration: 7700 / 10000 [ 77%]  (Sampling)
## Chain 1 Iteration: 7800 / 10000 [ 78%]  (Sampling)
## Chain 1 Iteration: 7900 / 10000 [ 79%]  (Sampling)
## Chain 1 Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1 Iteration: 8100 / 10000 [ 81%]  (Sampling)
## Chain 1 Iteration: 8200 / 10000 [ 82%]  (Sampling)
## Chain 1 Iteration: 8300 / 10000 [ 83%]  (Sampling)
## Chain 1 Iteration: 8400 / 10000 [ 84%]  (Sampling)
## Chain 1 Iteration: 8500 / 10000 [ 85%]  (Sampling)
## Chain 1 Iteration: 8600 / 10000 [ 86%]  (Sampling)
## Chain 1 Iteration: 8700 / 10000 [ 87%]  (Sampling)
## Chain 1 Iteration: 8800 / 10000 [ 88%]  (Sampling)
## Chain 1 Iteration: 8900 / 10000 [ 89%]  (Sampling)
## Chain 1 Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1 Iteration: 9100 / 10000 [ 91%]  (Sampling)
## Chain 1 Iteration: 9200 / 10000 [ 92%]  (Sampling)
## Chain 1 Iteration: 9300 / 10000 [ 93%]  (Sampling)
## Chain 1 Iteration: 9400 / 10000 [ 94%]  (Sampling)
## Chain 1 Iteration: 9500 / 10000 [ 95%]  (Sampling)
## Chain 1 Iteration: 9600 / 10000 [ 96%]  (Sampling)
## Chain 1 Iteration: 9700 / 10000 [ 97%]  (Sampling)
## Chain 1 Iteration: 9800 / 10000 [ 98%]  (Sampling)
## Chain 1 Iteration: 9900 / 10000 [ 99%]  (Sampling)
## Chain 1 Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1 finished in 0.2 seconds.
```

```
## Inference for Stan model: rt_cmdstanr_82ccf439e6ee2a6b08997cf2f7060755-202204071136-1-7bdd7e.
## 1 chains, each with iter=10000; warmup=5000; thin=1;
## post-warmup draws per chain=5000, total post-warmup draws=5000.
##
##          mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff
## P1       0.99    0.00 0.00     0.99     0.99     0.99     0.99     0.99  3783
## P0       0.99    0.00 0.00     0.99     0.99     0.99     0.99     0.99  3459
## rd       0.00    0.00 0.00    -0.01     0.00     0.00     0.00     0.00  3664
## rr       0.74    0.00 0.10     0.56     0.67     0.73     0.80     0.95  3649
## sr       0.26    0.00 0.10     0.05     0.20     0.27     0.33     0.44  3649
## lp__ -1321.40    0.02 1.01 -1324.12 -1321.77 -1321.08 -1320.68 -1320.41  2063
##      Rhat
## P1      1
## P0      1
## rd      1
## rr      1
## sr      1
## lp__    1
##
## Samples were drawn using NUTS(diag_e) at Thu Apr 07 11:36:57 2022.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
## ci_level: 0.8 (80% intervals)
## outer_level: 0.95 (95% intervals)
```



In class exercise

Aside from the "Aceh" trial, there are also two other ones being carried out. Try replacing "Aceh" with "Sarlahi" and "Sarlahi" to see if the results will be different.

References

Cinelli C, Pearl J. Generalizing experimental results by leveraging knowledge of mechanisms. Eur J Epidemiol. 2021 Feb;36(2):149-164. doi: 10.1007/s10654-020-00687-4. Epub 2020 Oct 18. PMID: 33070298.