BDA - Project

Leo Laitinen Marilla Malkki Otso Laasonen

Contents

1. Introduction

Tornadoes are rotating columns of air that have intrigued mankind for hundreds of years. They occur most numerous in the US where it is estimated that around 1200 tornadoes appear annually. They vary in size and intensity, with the largest, most extreme tornadoes reaching wind speeds of up to 480km/h.

In this report, we take a look at the relation between tornadoes and the injuries caused by them in the United States. We will be using two types of linear models whose distributions we approximate with Stans default sampler. (Hamiltonian Monte Carlo with No U-turn Sampler)

Next up we will describe the problem in more detail, and after that we will explain the dataset. The models are described in section 3, followed by converge diagnostics and posterior predictive checks. In chapter 10 we will move onto sensitivity analysis and then to model comparison. Finally, we discuss potential improvements and the conclusions we arrived to based on the results we obtained. The report concludes with self- reflection as its final chapter.

2. Data and the problem

We use Stan to see how are tornado injuries related to the magnitude of the tornado in the US. Additionally we analyzed whether the state the tornado was in had impact on the injuries. Data was obtained from 'kaggle.com', which provides free to use datasets for different purposes. The data contained a lot of extra columns that were unnecessary for our problem, and were hence trimmed. Naturally there are tons of analyses of tornadoes, as they have a huge impact on societies experiencing them. However we were unable to find any analysis using bayesian statistics for the relation of injuries and magnitude in the US.

Data

Our dataset contained information of tornadoes in the United States in years 1950-2021. We divided the dataset into groups by the state in which the tornado occurred. Then we calculated the average injuries caused by tornadoes of a certain magnitude in each state. The original data had 29 columns and 68829 rows. After the aforementioned calculation we were left with 3 variables of interest.

State

State tells the state in the US in which the tornado happened. This parameter functioned as the grouping in our hierarchical model.

Magnitude

This is the magnitude of the tornado, which was the covariate of our model.

Averaged injuries

Averaged injuries is a calculated variable that averages all the injuries by all tornadoes in a certain state and of certain magnitude.

3. Model

We decided to try linear regression in our analysis after transforming our dataset as described in the previous section. Our hypothesis was that the average of injuries was linearly proportional to the magnitude of a tornado.

We developed to models for the linear regression: a pooled model and a hierarchical model. The pooled model can be applied if the injuries are considered to be identically and independently distributed between the states. The hierarchical model allows the injuries to be dependent on the state of the tornado's occurrence.

Let x_{ij} be the magnitude of a tornado in state j and $\bar{y}_{ij}$ be the average of observed injuries caused by tornadoes of magnitude x_{ij} in certain state. The magnitude of a tornado can have values $\{0, 1, 2, 3, 4, 5\}$. However, tornadoes of magnitude 5 are very rare. Additionally, tornadoes occur in some states altogether only few times. We decided to filter the data such that we have the observations of the states where tornadoes of magnitude 0-4 have occurred at least once, which is totally 32 states.

Pooled model

Mathematically, the pooled model was defined as

$$\bar{y}_{ij} \sim \mathrm{N}(\alpha x_{ij} + \beta, \sigma)$$

$$\alpha \sim \mathrm{N}(0, 100)$$

$$\beta \sim \mathrm{N}(0, 100)$$

$$\sigma \sim \mathrm{N}(5, 100).$$

That is, the average injuries is normally distributed with a mean value which is a linear function of the magnitude x_{ij} of a tornado.

Hierarchical model

The mathematical definition of the hirearchical model was

$$\bar{y}_{ij} \sim \mathrm{N}(\alpha_j x_{ij} + \beta_j, \sigma)$$

$$\alpha_j \sim \mathrm{N}(\mu_\alpha, \sigma_\alpha)$$

$$\beta_j \sim \mathrm{N}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha \sim \mathrm{N}(0, 100)$$

$$\sigma_\alpha \sim \mathrm{N}(5, 100)$$

$$\mu_\beta \sim \mathrm{N}(0, 100)$$

$$\sigma_\beta \sim \mathrm{N}(5, 100)$$

$$\sigma \sim \mathrm{N}(5, 100).$$

Now for each state, we have own linear average parameters α_j and β_j which are defined by normal distribution with hyper parameters μ_α, μ_β, σ_α and σ_β. The hyper parameters are defined by normal hyper priors. Each of the states however have a common variance σ.

4. Priors

We decided to use weakly informative normal priors for each (hyper) parameter. We tested many different prior distributions and parameters and got decent performance with the chosen normal priors. Some of the other priors caused for example divergence in HMC.

We chose zero centered normal priors with a large standard deviation for parameters α, β, μ_α and μ_β. We believe that the true parameter value would be quite close to zero but the wide prior distribution allows the parameters to converge to afar from zero. For standard deviations σ, σ_α and σ_β, we chose normal priors which had the mean parameter at 5. We believe that there is some variance between the observations so the standard deviation would be larger than zero. Again, the standard deviation of the priors were chosen quite large such that we get a weakly informative prior.

5/6. Code

The Stan implementations of the models is shown below. The pooled model was implemented as

```
// Pooled linear model

data {
  int<lower=0> N;  // number of data points
  vector[N] x;// Magnitude of a tornado
  vector<lower=0>[N] y;// Injuries caused by a tornado
  real pmualpha;//prior mean of alpha
  real<lower=0> psalpha;//prior sd of alpha
  real pmubeta;//prior mean of beta
  real<lower=0> psbeta;//prior sd of beta
  vector[6] xpred;
}

parameters {
  real alpha;
  real beta;
  real<lower=0> sigma;
}

transformed parameters {
  vector[N] mu = alpha*x + beta;
}

model {
  alpha ~ normal(pmualpha, psalpha); //prior of alpha
  beta ~ normal(pmubeta, psbeta); //prior of beta
  sigma ~ normal(5, 100);
  y ~ normal(mu, sigma);
}
```

The hierarchical model is implemented as

```
data {
  int<lower=0> N;  // number of data points
  int<lower=0> J; // number of states
  vector[J] x[N];// Magnitude of a tornado
  vector<lower=0>[J] y[N];// Average injuries caused by a tornado
  vector[5] xpred;
}

parameters {
  real mualpha0;
  real<lower=0> salpha0;
  real mubeta0;
  real<lower=0> sbeta0;
  vector[J] alpha;
  vector[J] beta;
  real<lower=0> sigma;
}

transformed parameters {
  vector[J] mu[N];

  for (n in 1:N) {
    for (j in 1:J)
      mu[n, j] = x[n,j]*alpha[j]+beta[j];
  }
}

model {
  mualpha0 ~ normal(0, 100);
  salpha0 ~ normal(5, 100);;
  mubeta0 ~ normal(0, 100);
  sbeta0 ~ normal(5, 100);;
  sigma ~ normal(5, 100);

  for (j in 1:J) {
    alpha[j] ~ normal(mualpha0, salpha0); //prior of alpha
    beta[j] ~ normal(mubeta0, sbeta0); //prior of beta
    y[,j] ~ normal(mu[,j], sigma);
  }
}
```

The models were run with the script represented below

```
data_pooled <- readRDS(file='./data_pooled_ver2.Rda')
data_hx <- readRDS(file='./data_hierarchical_x_ver2.Rda')
data_hy <- readRDS(file='./data_hierarchical_y_ver2.Rda')

# Fit pooled model
pooled_model <- cmdstan_model(stan_file = "pooled_linear_model.stan")
stan_data <- list(
  N = nrow(data_pooled),
  x = data_pooled$x,
  y = data_pooled$y,
  pmualpha = 0,
  psalpha = 300,
  pmubeta = 0,
  psbeta = 100,
  xpred = c(0,1,2,3,4,5)
)
fit_pooled <- pooled_model$sample(data = stan_data, refresh=1000)

# Fit hierarchical model
hierarchical_model <- cmdstan_model(stan_file = "hierarchical_linear_model.stan")
stan_data <- list(
  N = nrow(data_hy),
  J = ncol(data_hy),
  x = data_hx,
  y = data_hy,
  xpred = c(0,1,2,3,4)
)
fit_hierarchical <- hierarchical_model$sample(data = stan_data, refresh=1000)
```

That is, both models were run in Stan such that they had 4 chains with chain length of 2000 and warm-up length of 1000. With these options, we get overall 4000 draws from the distributions.

```
data_pooled <- readRDS(file='data_pooled_ver2.Rda')
data_hx <- readRDS(file='data_hierarchical_x_ver2.Rda')
data_hy <- readRDS(file='data_hierarchical_y_ver2.Rda')
```

Pooled model:

```
pooled_model <- cmdstan_model(stan_file = "pooled_linear_model.stan")
stan_data <- list(
  N = nrow(data_pooled),
  x = data_pooled$x,
  y = data_pooled$y,
  pmualpha = 0,
  psalpha = 100,
  pmubeta = 0,
  psbeta = 1000,
  xpred = c(0, 1, 2, 3, 4)
)
```

```
fit_pooled <- pooled_model$sample(data = stan_data, refresh=1000)

draws_p <- fit_pooled$draws()
```

Hierarchical model:

```
hierarchical_model <- cmdstan_model(stan_file = "hierarchical_linear_model.stan")
stan_data <- list(
  N = nrow(data_hy),
  J = ncol(data_hy),
  x = data_hx,
  y = data_hy,
  pmualpha = 0,
  psalpha = 100,
  pmubeta = 0,
  psbeta = 1000,
  xpred = c(0, 1, 2, 3, 4)
)
fit_hierarchical <- hierarchical_model$sample(data = stan_data, refresh=1000)

draws_h <- fit_hierarchical$draws()
```

7. Convergence diagnostics

First, $\hat{R}$ analysis from the summaries of the models. The values are good is they're close to 1 (< 1.01). ($\hat{R}$ values are retrieved from the stan model with summary(). The computation is done with the newer, rank-based method proposed in Vehtari et al. (2021))

Pooled:

```
summary_p <- fit_pooled$summary()
summary_p[1:4,c("variable", "rhat")]
```

```
## # A tibble: 4 x 2
##   variable  rhat
##   <chr>    <dbl>
## 1 lp__      1.00
## 2 alpha     1.00
## 3 beta      1.00
## 4 sigma     1.00
```

Based on the values in the pooled model summary (rhat is smaller than 1.01), the variance between chains is very small in the pooled model and the chains likely converge.

Hierarchical:

```
summary_h <- fit_hierarchical$summary()
print(summary_h[1:10,c("variable", "rhat")], n=10)
```

```
## # A tibble: 10 x 2
##    variable  rhat
##    <chr>    <dbl>
##  1 lp__      1.03
##  2 mualpha0  1.00
##  3 salpha0   1.00
##  4 mubeta0   1.03
##  5 sbeta0    1.03
##  6 alpha[1]  1.00
##  7 alpha[2]  1.01
##  8 alpha[3]  1.00
##  9 alpha[4]  1.00
## 10 alpha[5]  1.00
```

```r
print(summary_h[65:70,c("variable", "rhat")], n=6)
```

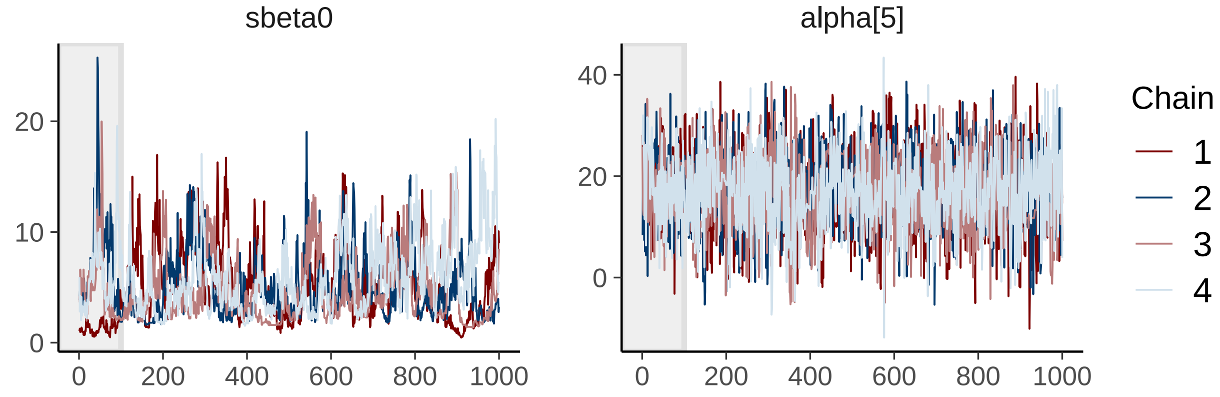
```
## # A tibble: 6 x 2
##   variable  rhat
##   <chr>    <dbl>
## 1 beta[28]  1.02
## 2 beta[29]  1.02
## 3 beta[30]  1.01
## 4 beta[31]  1.02
## 5 beta[32]  1.02
## 6 sigma     1.00
```

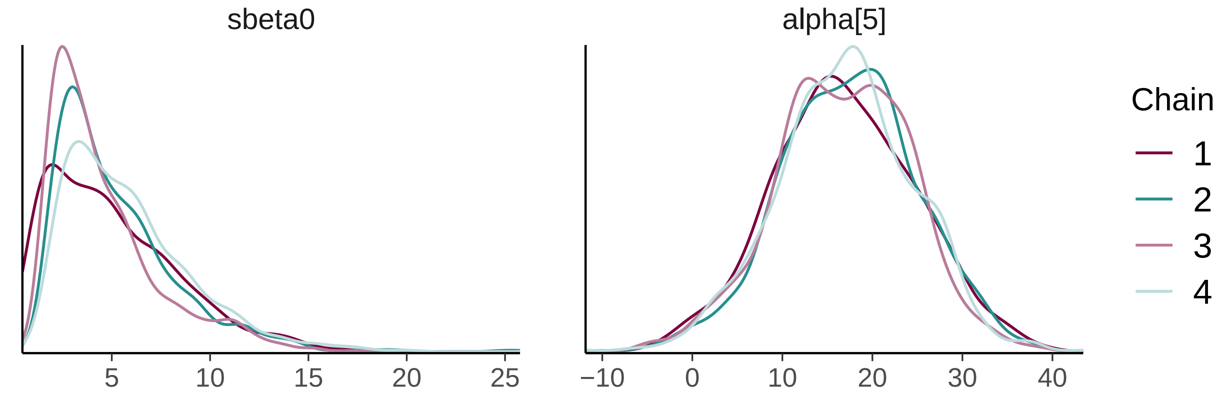
Many rhat values in the hierarchical model's summary are close to the threshold (1.01), but many are above it. This suggests that the chains might not converge and they sample slightly different distributions. Further analysis is needed.

The chains convergence issues can be visualized with traceplots. Sbeta0 and alpha[5] are shown as examples. It's also possible to look at how the distributions look for different chains. Depending on the run, they resemblance changes, but the point is that the distributions of each chain can look visibly different.

```r
color_scheme_set("mix-blue-red")
trace <- mcmc_trace(draws_h, pars = c("sbeta0", "alpha[5]"), n_warmup = 100)
color_scheme_set("mix-teal-pink")
overlay <- mcmc_dens_overlay(draws_h, pars = c("sbeta0", "alpha[5]"))

grid.arrange(trace, overlay, nrow = 2)
```





Next, the effective sample sizes (ESS, or n_eff) for both models. This information is also retrieved from with summary(). As can be seen from the tables, stan actually produces two ESS values, ess_tail for for the sampling efficiency in the tails of the posterior and ess_bulk for sampling efficiency in the bulk of the posterior. These are used since they provide more information than a generic ESS and are easily available.

ESS for pooled model:

```
summary_p[1:4,c("variable", "ess_bulk", "ess_tail")]
```

```
## # A tibble: 4 x 3
##   variable ess_bulk ess_tail
##   <chr>       <dbl>    <dbl>
## 1 lp__        1654.    1853.
## 2 alpha       1441.    1723.
## 3 beta        1453.    1637.
## 4 sigma       2303.    1986.
```

The pooled model's ESS values are good, all above 1500. This supports confidence in the sampling, since it means that the effective sample size is large and the draws are used.

ESS for hierarchical model:

```
print(summary_h[1:10,c("variable", "ess_bulk", "ess_tail")], n=10)
```

```
## # A tibble: 10 x 3
##    variable ess_bulk ess_tail
##    <chr>       <dbl>    <dbl>
##  1 lp__         106.     55.6
##  2 mualpha0     780.   1474.
##  3 salpha0     1426.   1573. 
##  4 mubeta0      169.     79.2
##  5 sbeta0       108.     58.2
##  6 alpha[1]     808.   1336. 
##  7 alpha[2]     659.    306. 
##  8 alpha[3]     969.   1117. 
##  9 alpha[4]    1993.   1871. 
## 10 alpha[5]    1391.   2121. 
```

```
print(summary_h[65:70,c("variable", "ess_bulk", "ess_tail")], n=6)
```

```
## # A tibble: 6 x 3
##   variable ess_bulk ess_tail
##   <chr>       <dbl>    <dbl>
## 1 beta[28]     270.     75.3
## 2 beta[29]     283.     91.6
## 3 beta[30]     291.     99.6
## 4 beta[31]     255.     89.4
## 5 beta[32]     297.    102. 
## 6 sigma        975.   1147. 
```

The hierarchical model's ESS values are also satisfactory, according to the diagnostics. It can be noted, that the EES values for alphas seem to be clearly larger than the EES values of betas.

Especially sbeta0 (a hyperparameter) seems to have smaller EES and it also had the largest rhat value. While the ESS is still described as satisfactory, together with the rhat it does suggest there is a small issue with the sampling.

Then, divergences and tree depth. The diagnostics of the models are retrieved by cmdstan_diagnose(). As the Rhat values and ESS were already discussed, it's sufficient to just look at the first two sections.

Pooled model: Based on the diagnostics below, there are no divergences and all treedepths are small enough.

```
fit_pooled$cmdstan_diagnose()
```

```
## Processing csv files: /tmp/RtmpNJAfL4/pooled_linear_model-202212041941-1-6b3dc6.csv, /tmp/RtmpNJAfL4/
## 
## Checking sampler transitions treedepth.
## Treedepth satisfactory for all transitions.
## 
## Checking sampler transitions for divergences.
## No divergent transitions found.
## 
## Checking E-BFMI - sampler transitions HMC potential energy.
## E-BFMI satisfactory.
## 
## Effective sample size satisfactory.
## 
## Split R-hat values satisfactory all parameters.
## 
## Processing complete, no problems detected.
```

Hierarchical model: The diagnostics below show that treedepths are fine, but there are some divergences (about 7% of transitions). This suggest that there might be a problem with HMC being unable to fully explore the posterior distribution. Taking into account the larger rhat values, it's likely the chains in this model don't converge fully.

```
fit_hierarchical$cmdstan_diagnose()
```

```
## Processing csv files: /tmp/RtmpNJAfL4/hierarchical_linear_model-202212041941-1-4eac01.csv, /tmp/Rtmp
##
## Checking sampler transitions treedepth.
## Treedepth satisfactory for all transitions.
##
## Checking sampler transitions for divergences.
## 227 of 4000 (5.67%) transitions ended with a divergence.
## These divergent transitions indicate that HMC is not fully able to explore the posterior distributio
## Try increasing adapt delta closer to 1.
## If this doesn't remove all divergences, try to reparameterize the model.
##
## Checking E-BFMI - sampler transitions HMC potential energy.
## The E-BFMI, 0.17, is below the nominal threshold of 0.30 which suggests that HMC may have trouble ex
## If possible, try to reparameterize the model.
##
## Effective sample size satisfactory.
##
## Split R-hat values satisfactory all parameters.
##
## Processing complete.
```

8. Posterior predictive check

We generated predictive values from the posterior distributions for both models. To make a predictive check, we compare the histograms of the real observed data and predictive data with each magnitude value separately. The observed values and generated draws are collected to data frames `df_y`, `df_p` and `df_h`. The histograms are presented below

```
yhist <- mcmc_hist(df_y, pars = c('y0', 'y1','y2','y3','y4'))
phist <- mcmc_hist(df_p, pars = c('y0', 'y1','y2','y3','y4'), binwidth = 30)
hhist <- mcmc_hist(df_h, pars = c('y0', 'y1','y2','y3','y4'), binwidth = 30)
grid.arrange(yhist, phist, hhist, nrow = 3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

As we can see from the plots, the observed data does not have the shape of a normal distribution. However, we can make some deduction of the predictive draws with these histograms. As we can see, the average injuries increase as the magnitude of tornadoes increase. With the smaller magnitudes, the observed injuries were close to zero on average. The predictive draws for both models have the mean value also quite close to zero when the magnitude was small. With magnitudes 3 and 4, the injuries are evidently greater than zero on average by the observations. Again, the models have predicted this and the mean of the distribution is clearly larger.

Injuries caused by a tornado is a quantity that has only non-negative values. Our predictions however have also negative draws for average injuries. Therefore, we can say that the predictions are not quite optimal with this model.

9. Predictive performance assessment

Predictive performance assessment wasn't conducted for the models because the goal isn't classifying so accuracy wouldn't make any sense. Splitting the data to training and testing was considered (to compute MSE for predicted values) but decided against it. The idea is to model the number of injuries and since the

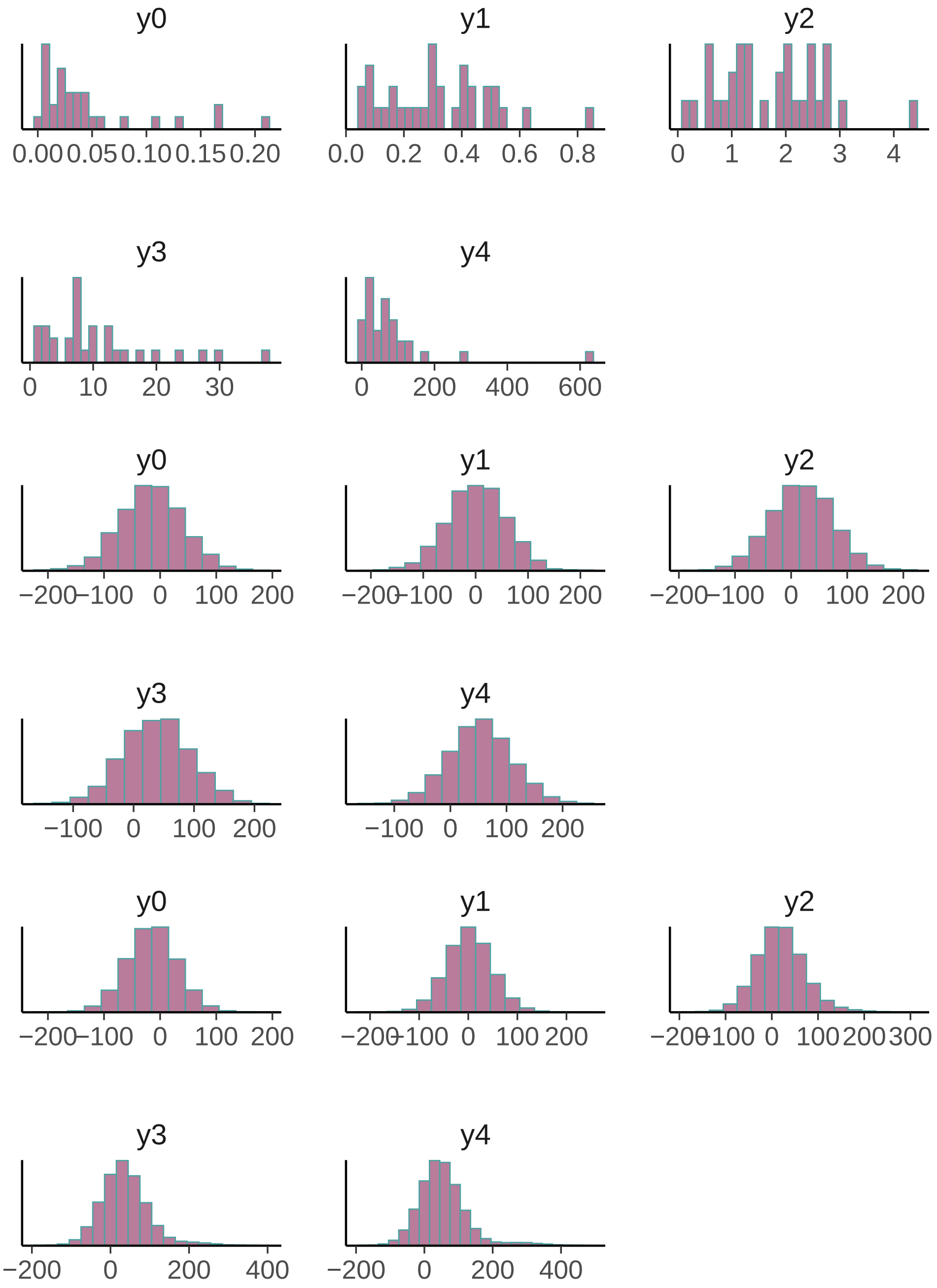


Figure 1: Histograms of real observations, draws from pooled model and draws from hierarchical model for each tornado magnitudes. First two rows are the observed values for tornadoes of magnitude 0-4. Similarly, third and fourth row contain the draws of the pooled model and fifth and sixth row contain the draws of the hierarchical model for tornadoes of magnitude 0-4.

linear models don't fit the data very well, getting figures for MSE wouldn't have provided more value to the analysis.

10. Sensitivity analysis with respect to prior choices

First the sensitivity to priors is tested for the pooled model. Three different priors are tested:

1. Uniform priors for alpha and beta: $uniform(0, 100)$

2. Inv_chi_square for sigma: inv_chi_square(0.1)

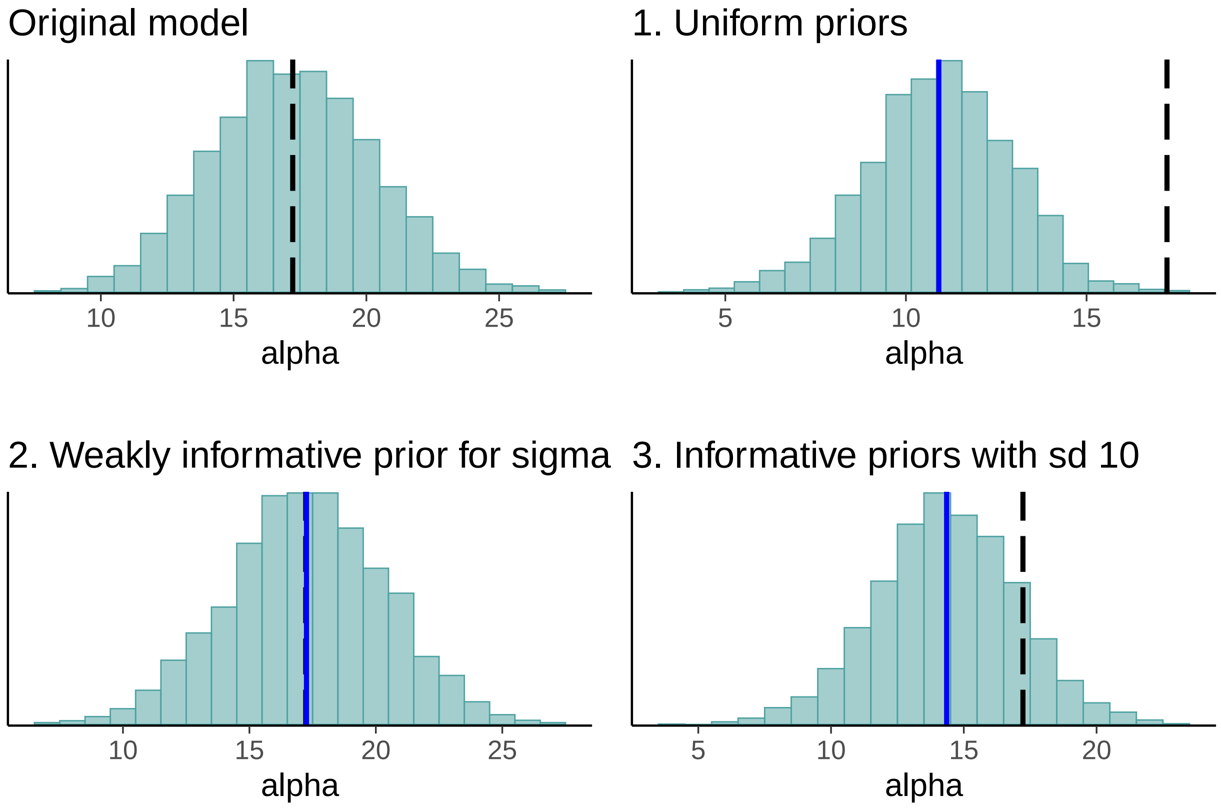
3. Smaller sd for alpha and beta: $normal(mean, 10)$ instead of the previous 100 and 1000

Other values stay the same when priors are changed.

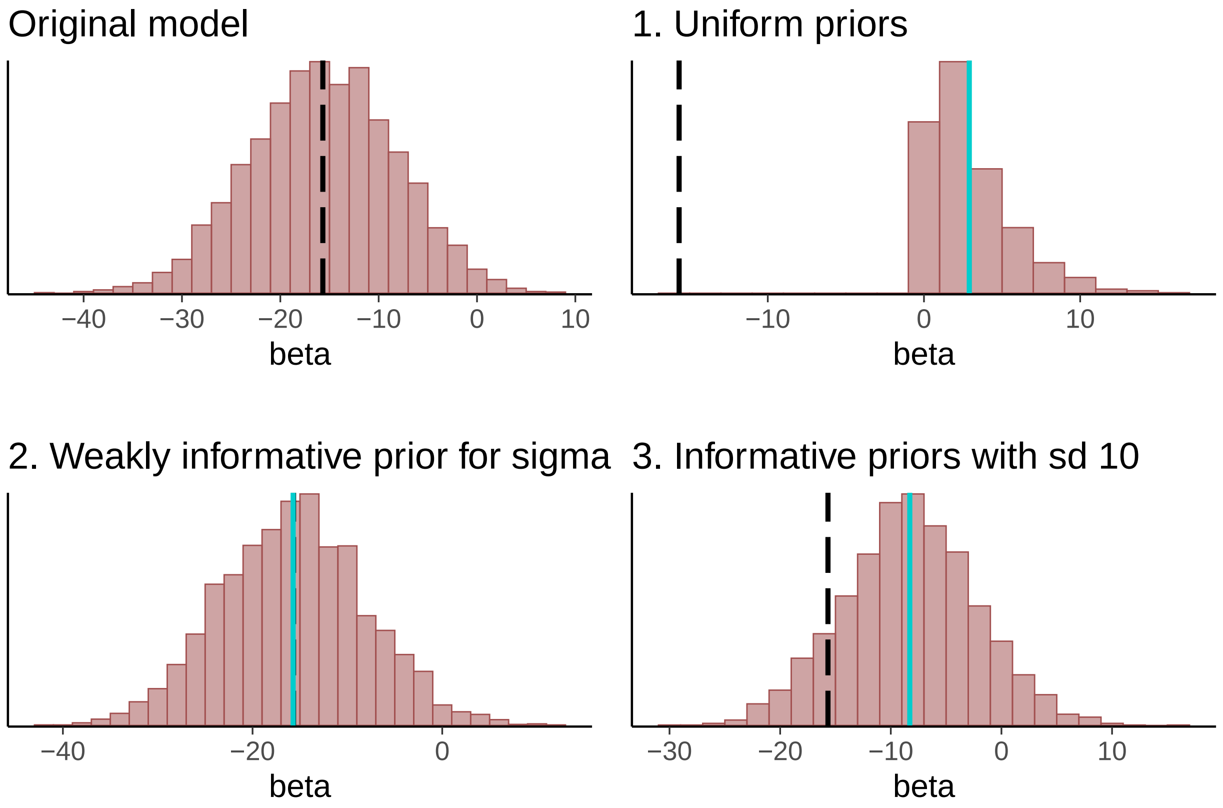
Below the models are fitted and sampling is done. Then the new draws are used to see how the test prios affect alpha and beta. Plots are shown for all.

Three pooled models are fitted with the new priors, code hidden for readability.

The plots below visualize how the priors affect the distribution of alpha, the dotted line showing the original model's mean in all plots, while the blue displays the new one.



These plots show the changes to the distribution of beta. Once again, the dotted line shows the mean of beta in the original model while the teal shows the new mean.



From the plots, it seems that the pooled model isn't very sensitive to different priors if the other prior is weakly informative (like the 2. prior). Changing from weakly informative to uniform or very restricted sd (1. and 3.) has a clear effect: making the mean of alpha lower and the mean of sigma higher.

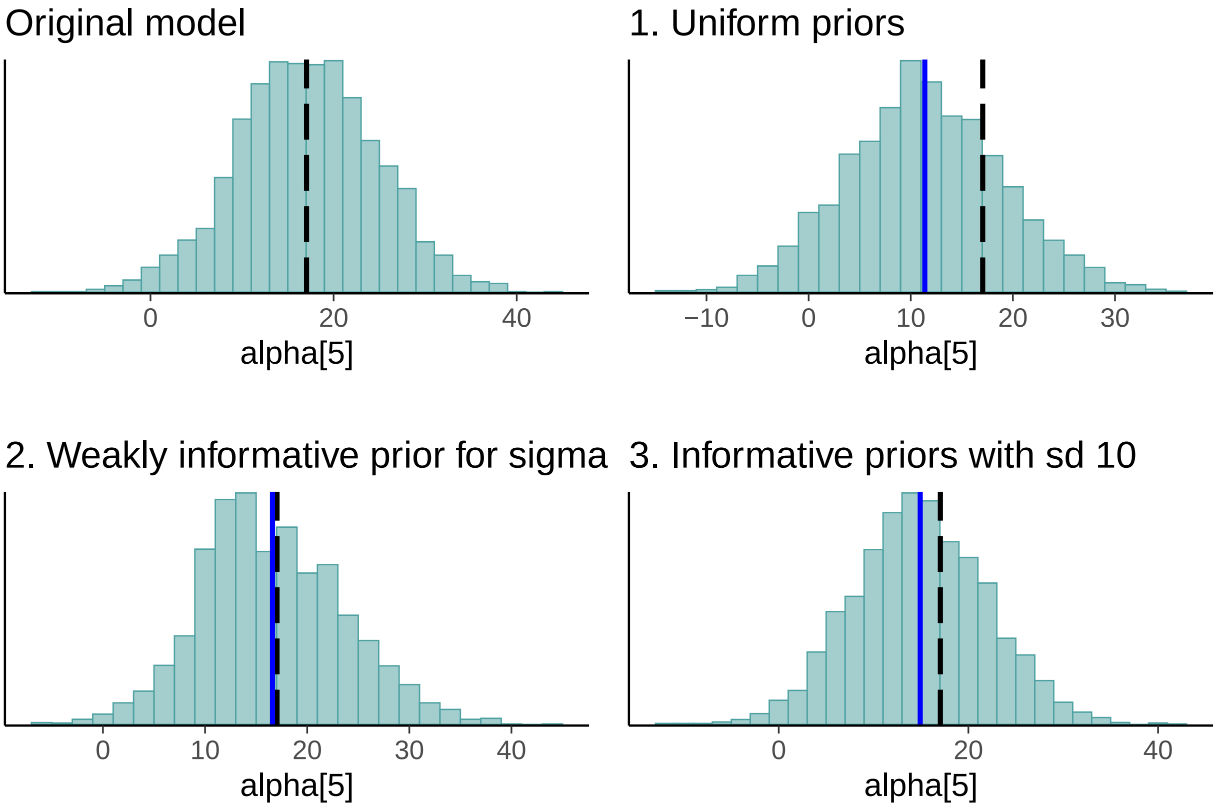
Next, sensitivity to priors for the hierarchical model. Since there are so many alphas and betas in the hierarchical model (32 states in the data, each having their own), sensitivity to priors will be visualized through two randomly selected ones: alpha[5] and beta[21]. We assume that they are representative in how the priors affect the results in general.

Again, three different priors are tested:

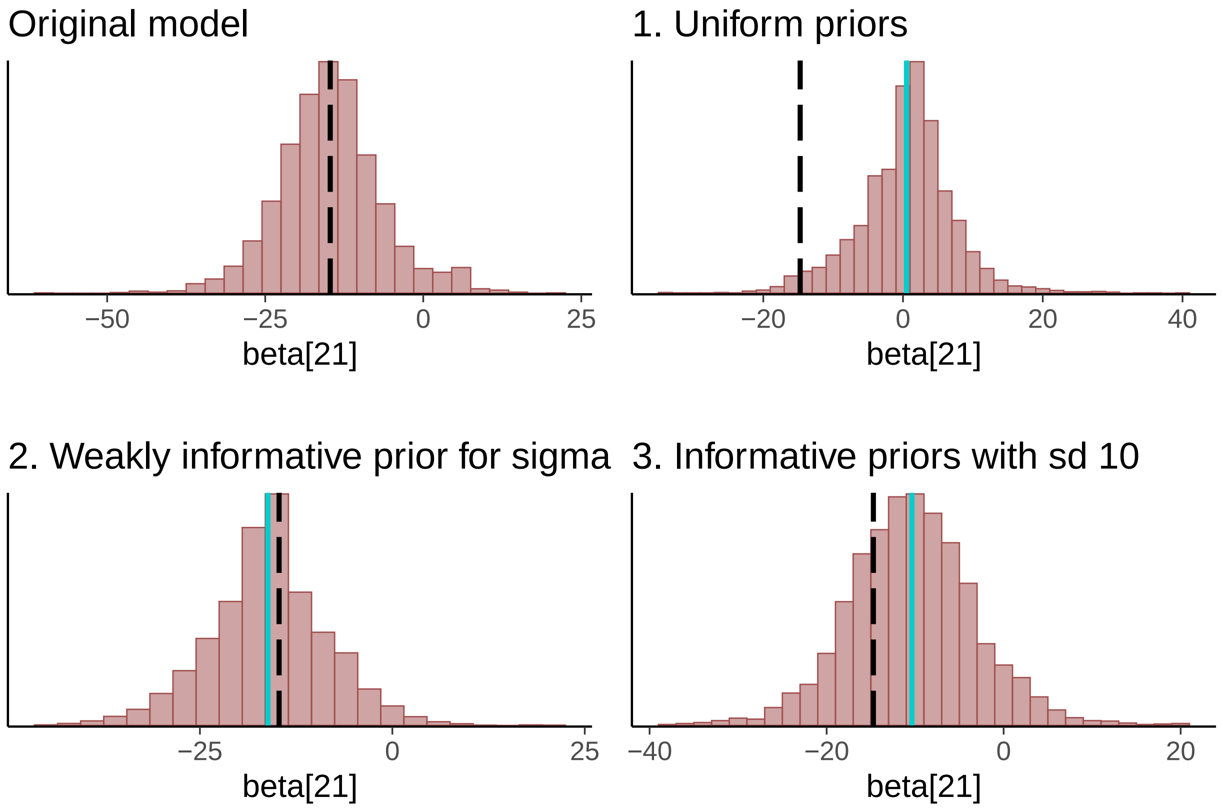
1. Uniform priors for hyperparameters of alpha and beta: $uniform(0, 100)$
2. Inv_chi_square for sigma: inv_chi_square(0.1)
3. Smaller sd for hyperparameters of alpha and beta: $normal(mean, 10)$ instead of the previous 100

Three hierarchical models are fitted with the new priors, code hidden for readability.

The plots below visualize how the priors affect the distribution of alpha[5].



These plots show the changes to the distribution of beta[21].



From the plots, it can be seen that the hierarchical model is perhaps a bit less sensitive to priors in general. Of course, here only priors of hyperparameters were changed. Again, the weakly informative prior (2.) had very little effect. Uniform and smaller sd did affect the means and distributions clearly, so the hierarchical model is also somewhat sensitive to the choice of prior.

11. Model comparison with LOO-CV

Pooled model: PSIS-LOO elpd values and the $\hat{k}$-values

```
loo_pooled <- fit_pooled$loo()
frame <- as.data.frame(loo_pooled$estimates)
elpd <- frame["elpd_loo",]
elpd
```

```
##             Estimate       SE
## elpd_loo -889.3912 76.20991
```

```
pareto_k_table(loo_pooled)
```

```
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     159   99.4%   569
##  (0.5, 0.7]   (ok)         0    0.0%   <NA>
##    (0.7, 1]   (bad)        0    0.0%   <NA>
##    (1, Inf)   (very bad)   1    0.6%   1
```

So the pareto k values can be seen from the table, the elpd is roughly: -890

Hierarchical model: PSIS-LOO elpd values and the $\hat{k}$-values

```
loo_hierarchical <- fit_hierarchical$loo()
frame <- as.data.frame(loo_hierarchical$estimates)
elpd <- frame["elpd_loo",]
elpd
```

```
##             Estimate       SE
## elpd_loo -868.6938 53.67754
```

```
pareto_k_table(loo_hierarchical)
```

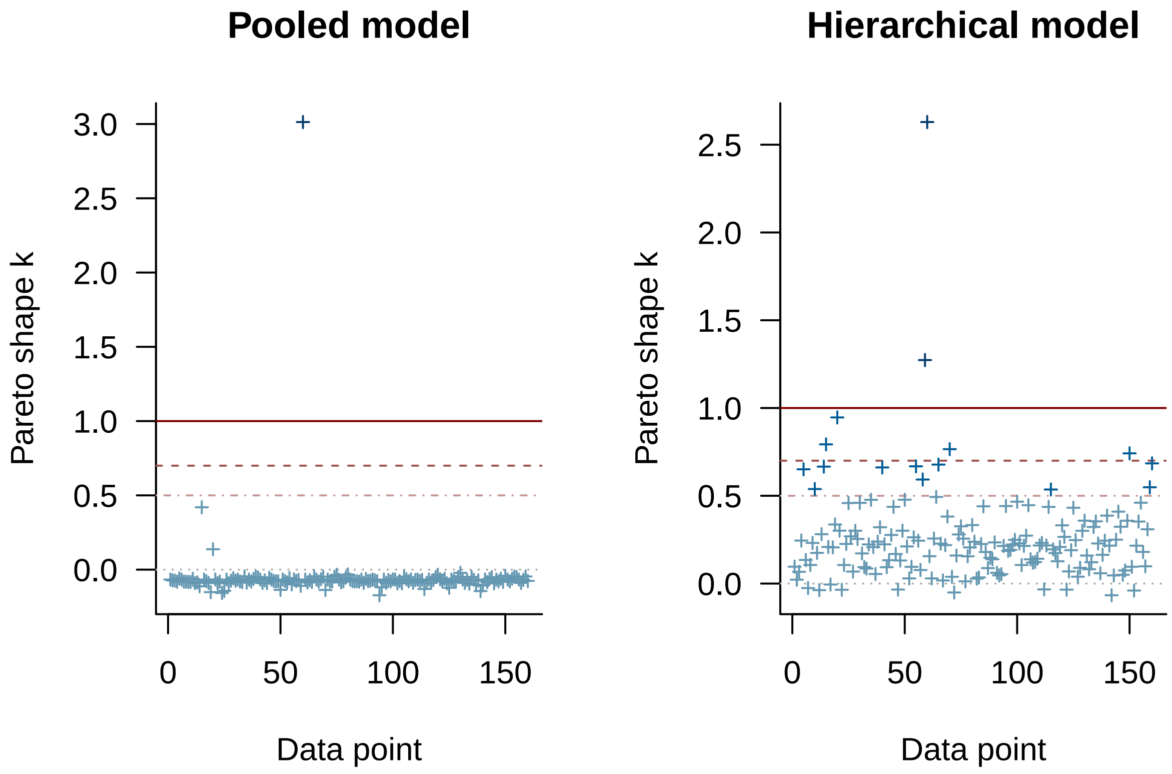
```
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     144   90.0%   285
##  (0.5, 0.7]   (ok)        10    6.2%   99
##    (0.7, 1]   (bad)        4    2.5%   22
##    (1, Inf)   (very bad)   2    1.2%   1
```

So the pareto k values can be seen from the table, the elpd is roughly: -870

As can be seen from the tables and the plots, the pooled model has generally very good pareto k values (below 0.5). This means that the PSIS-LOO estimate is reliable.

The hierarchical model is worse in this regard, having mostly good and ok values. It does also have some bad and very bad values (values above 0.7 and 1), which suggests that the PSIS-LOO models for the hierarchical model might be too optimistic.

Visualization:



Finally compare the models based on the PSIS-LOO values:

```
comp_p_to_h <- loo_compare(loo_pooled, loo_hierarchical)
comp_p_to_h
```

```
##        elpd_diff se_diff
## model2   0.0       0.0  
## model1 -20.7      28.7  
```

The results show that model2, meaning the hierarchical model, has higher PSIS-LOO and thus should be selected if it were the only thing to consider. However, the pareto k values make the reliability of the PSIS-LOO estimate for the hierarchical model questionable. Considering that this model also had some convergence issues, the pooled model might be a better choice.

12. Discussion of issues and potential improvements

One of the biggest pitfalls of our model was the lack of datapoints for our hierarchical model. Each state had only one average calculation per magnitude, and some magnitudes had no datapoints at all. (Set to zero in that case) This problem naturally did not occur the same way in pooled, hence why chains converged much better with that model. The relation between the magnitude and the injuries is not very linear, hence why linear model struggles with the predictions.

A better formulation of our data would have perhaps functioned better than the this one. Some model that would have given more datapoints per grouping could have gotten more out of it. That could have potentially fixed the linearity problem as well.

13. Conclusion

We made two linear regression models that were applied in predictions of average injuries caused by a tornado. We noticed that the simple linear regression models were not fully able to predict the injuries. Extreme phenomena needs extremely well developed models for proper predictions.

However, we made two working models which could predict some of the qualities of the problem. For example, the linear model predicted that the injuries increase as the magnitude of a tornado grows. Therefore with a little development, the model could probably make somewhat plausible predictions.

14. Self-reflection

We learned that data analysis is a hard topic to excel at. The key of success is to find proper mathematical models such as correct probability distribution to represent the data. Additionally, the choice of priors, hierarchy in models and covariate selection can major vast effect on the results obtained in the analysis. We believe that the injuries caused by a tornado could be predicted properly with a better model. Thus, research in tornadoes and bayesian inference should be continued.

Otso

Sometimes models just don't work the way you expect them to. There are so many parameters and variables that can be used and considered that an exhaustive analysis is rather difficult.

Marilla

I learned a lot about plotting with R and how to visualize diagnostics for the models.

Leo

I learned coding in R and developing of Bayesian models in Stan.