STAT 447C Project: Bayesian Models on Monthly Earthquakes Counts

Jerry Gai, Sunny Lau

1. Introduction

Earthquake forecasting has been a critical challenge in geophysical risk assessment. It is essential to model temporal patterns of earthquakes in regions that is seimically active, including Northern California, for hazard preparedness and early-warning strategies. Since traditional frequentist models mostly rely on point estimates, which fail to capture uncertainty, this project aims to apply Bayesian statistical methods to model monthly counts of M>=4.0 earthquakes in Northern California based on the dataset from Northern California Earthquake Data Center (NCEDC) (NCEDC, 2014).

Our goal is to construct and compare time-series and state-space models that can capture the underlying dynamic latent processes which drive seismic activity. The first model is a Bayesian Poisson state-space model with time-varying log-intensity via an autoregressive latent process. To capture and reflect the flexibility, another Negative Binomial model based on smaller regions that accounts for extra dispersion and seasonality, is also implemented and compared in terms of their prediction accuracy. For both models, prior explanation, posterior inferences, posterior checks, and forecasting will be performed.

Jerry works on the Negative Binomial model, and Sunny works on the Poisson model. We explore the datasets, compare the models, and complete the report collaboratively. All R source code, Stan models, data sets, and analysis scripts are available in our public GitHub repository:
https://github.com/GerGerGai/Bayesian_TimeSeries_StateSpace_Modeling

2. Problem Formulation In Details

The real-world inference problem that this project aims to solve is modelling and forecasting monthly earthquake counts, with the specific theme of “time series and state-space models”, mainly in the region of Northern California (NC).

The reason for choosing NC is that it is one of the most seismically active regions in North America, and it is located on the boundary between the North American Plate and the Pacific Plate, which is well-known as the San Andreas Fault. Since the seismic activity is frequent here with mostly small- to medium-magniude events and occasionally large and destructive, this region is ideal for studying dynamics of seismology and building models with the consideration of temporality.

Focusing on M4+ earthquakes balances data reliability and relevance, since these earthquakes are usually felt by people, which often causes minor damages, or even is classified as foreshock or aftershock of larger events. Modelling this magnitude class allows us to analyze informative temporal patterns while avoiding excessive noise from microseismicity. On top of that, it serves for several practical purposes, including seismic hazard assessment, early-warning system calibration, operational earthquake forecasting, and infrastructure and risk planning.

This problem is formulated as inference on a latent dynamic rate process. For each month t, we model the observed count y_t from Poisson or Negative Binomial distribution with a latent log-intensity $\log \lambda_t$ or $\log \mu_t$. For the Poisson model, the rate evolves via first-order autoregressive (AR(1)) process. For the negative binomial model, it allows overdispersion and seasonal structure through hierarchical coefficients and variance modeling.

This setup fits with the theme “time series and state-space models.” It allows us to capture both temporality and stochasticity in earthquake activity and evaluate each model’s ability to represent and predict real seismic behavior.

3. Literature Review

Before introducing our models, some major characteristics of earthquake statistics should be explored first. A foundational contribution by Kagan (2010) systematically examined the statistical distributions of earthquake counts, emphasizing four major features: (1) earthquake counts are temporally dependent, often exhibiting

autoregressive behavior; (2) the data are typically overdispersed, with sample variance exceeding the mean—violating the equidispersion assumption of the Poisson distribution; (3) seismicity varies across geographic regions, necessitating localized or hierarchical models; and (4) earthquakes exhibit clustering, both temporally (e.g., aftershock sequences) and spatially, especially following large-magnitude events. These observations directly motivate our decision to use autoregressive components in the Poisson model and hierarchical and overdispersed structures in the Negative Binomial (NegBin) model. While Kagan focuses more on theoretical justification for statistical distributions, our implementation takes these insights and develops practical Bayesian models that address these statistical challenges in an applied forecasting setting.

We also draw inspiration from Natvig and Tvete (2004, 2007), who developed Bayesian hierarchical space–time models for earthquake analysis. Their models incorporate both spatial and temporal random effects. However, their goal is to predict maximum magnitude per region and time interval, whereas we focus on forecasting monthly counts of magnitude ≥ 4 earthquakes. Additionally, our NegBin model includes covariates such as depth, lagged magnitude, and location, and models overdispersion with observation-level predictors (Nst, RMS, CLO), allowing for more flexible variance modeling.

Our Negative Binomial model structure is also partially influenced by the flexible dispersion formulation strategies used in mixture-like count modeling. Although we do not implement an explicit finite mixture model, our use of per-observation dispersion modeled via covariates mimics mixture behavior, making the model more adaptive to rare, extreme-count observations.

There are also some other literature regarding earthquakes science that we reviewed for selecting certain variables and modeling techniques and we have included them as in-text citations as we explain our models further below.

In summary, our work builds on prior theoretical and applied studies in seismology and Bayesian modeling, extending them in a novel direction: we apply hierarchical Bayesian count models to monthly earthquake count prediction, integrate spatial and temporal covariates, and explore overdispersion and tail behavior in a unified framework not previously used on this dataset.

4. Data Analysis - Simple Poisson Model

4.1 Model specification:

Priors:

$$\alpha \sim N(0, 1)$$
$$\phi \sim \text{Uniform}(-1, 1)$$
$$\sigma \sim \text{half-Cauchy}(0, 1)$$
$$z_t \sim N(0, 1)$$
$$\theta_{lagmag}, \theta_{depth} \sim N(0, 0.5)$$

Likelihood:

$$y_t | \lambda_t, \theta \sim \text{Poisson}(\exp(\eta_t))$$

where:

$$\eta_t = \log \lambda_t + \theta_{depth} * \text{depth} + \theta_{lagmag} * \text{lagmag}$$
$$\log \lambda_t = \alpha + \phi * (\log \lambda_{t-1} - \alpha) + \sigma * z_t \text{ for t } \geq 2$$
$$\log \lambda_1 = \alpha + \sigma * z_1$$

4.2 Prior choices and parameters interpretations

While all priors are weakly informative, they are domain-aware. The interpretations of parameters are as follows:

α:. The “background” log-rate when predictors are 0 and the process is at equilibrium. The baseline rate is between 0.3 and 7 earthquakes per month.

ϕ: Month-to-month persistence, representing AR(1) process

σ: Volatility of rate changes, and half-Cauchy prior shrinks extreme values (Gelman, 2006). This allows for medium to large volatility.

θ_{depth}, θ_{lagmag}: Effect of average depth on counts and average magnitude in the previous month, respectively.

λ_i: instantaneous (latent) mean monthly count

The reason for this non-centred parameterization is that if we centred $\log \lambda_{t-1}$ for sampling $\log \lambda_t$, the posterior correlations between consecutive states are very strong, so MCMC may get stuck. Therefore, writing it as a deterministic function of $\alpha, \phi, \sigma, z_t$ with independent standard-normal z helps eliminating the funnel-shaped posterior geometry as shown in Stan User Guide, and ESS is then increased for λ_t, σ.

4.3 Posterior Dignostics and Predictive Checks

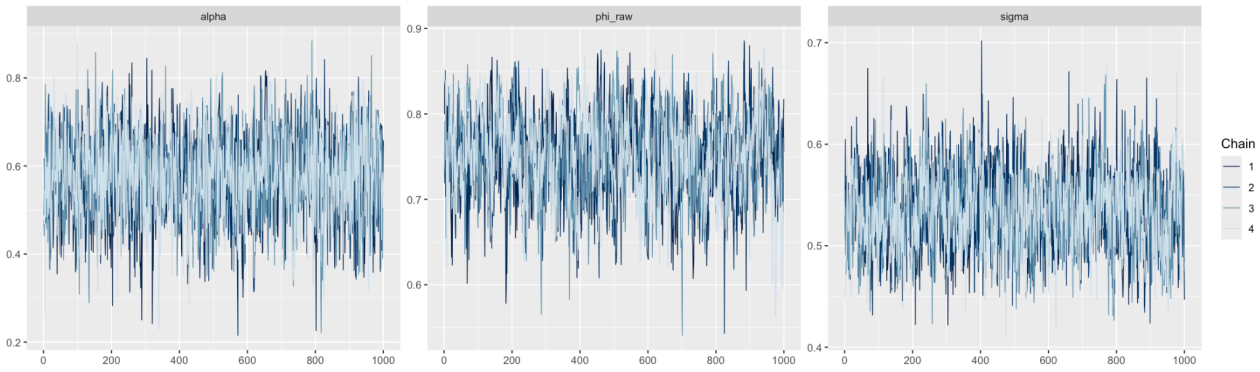

Figure 1: Trace plots for α, ϕ, σ (left to right respectively)

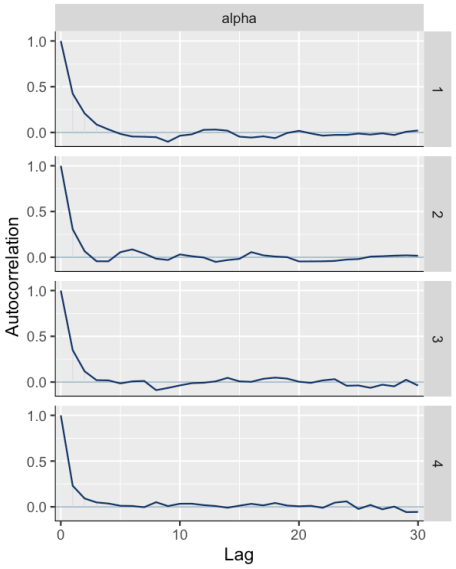

Fig. 2: Autocorrelation plot for α

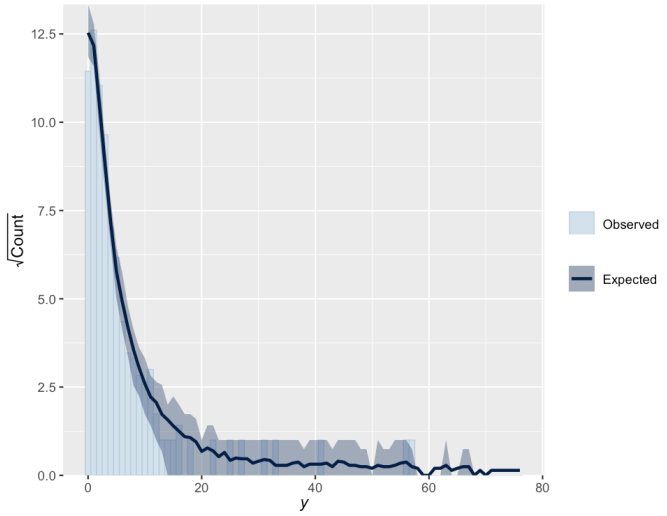

Fig. 3: Rootogram for posterior predictive check

For this model, all chains mixed well as seen in trace plots Fig. 1, and $\hat{R}$ < 1.01 and effective size > 400 for every parameter. There are no divergent transitions and tree-depth saturations reported. There is also no long-term autocorrelation for α as shown in Fig. 2. For posterior predictive check, the rootogram in Fig. 3 shows that posterior predictive samples capture the mean trend well but overestimate counts of zero earthquakes and underestimate some high counts, which suggests underdispersion. We also computed pointwise 95% predictive credible interval for y_{rep} and 99.8% of observed counts fell inside, suggesting an adequate coverage but possibly overconservative.

4.4 Forecasting and Out-of-Sample Evaluation

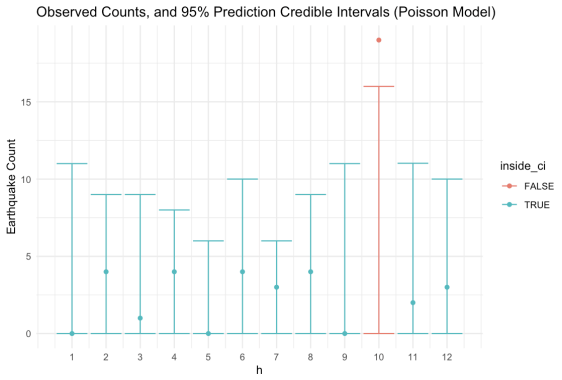

Fig. 4: Forecast result: Observed Counts, and 95% Prediction Credible Intervals (Poisson Model)

	RMSE	MAE
Poisson Model	4.67	2.79

Table 1: Metrics for evaluating the prediction result of the Poisson Model

To test the predictability of our model, we hold out the last 12 months (2023/03 ~ 2024/02) data for testing and forecasting. The result is not very good, with 91.6% of observed counts falling into 95% prediction credible intervals. The overall RMSE is 4.67 counts/month, MAE is 2.79 counts per month, suggesting a small error with these prediction results.

5. Data Analysis - Advanced Negative Binomial Model

5.1 Model Explanation

We model the monthly counts of earthquakes with magnitude >= 4, using a Bayesian hierarchical negative binomial regression model. This enables:

- Mean trends in seismic activity over time, space, and conditions.
- Varying dispersion depending on data quality metrics.
- Implicit mixture-like behavior to address extreme tail issues.

Earthquake data is grouped by region and month with each region defined by dividing the space into 2*2 degree latitude-longitude grids. For each region-month group, we compute:

- Count of earthquakes: y_i
- Average covariates: Depth, Latitude, Longitude, and lagged magnitude
- Time-based features: A continuous time index and sin/cos terms for seasonality.
- Data quality metrics: Nts (stations counts), RMS (residual), and CLO (coverage loss).

Even though we didn't explicitly use a classical mixture model, the dispersion formulation behaves like a continuous mixture:

- Every observation y_i has its own variance ϕ_i, which is equivalent to sampling from a different Negative Binomial distribution per observation.
- So in effect, the model is a mixture over infinitely many NB components, each shaped by its data quality indicators.

5.2 Model specification

Priors:

1. Region-level effects

$\alpha_{ri} \sim N(0, 2)$

$\beta^{time}_{ri} \sim N(0, 2)$

$\beta^{sin}_{ri} \sim N(0, 2)$

Likelihood

1. Observation model:

$y_i \sim NegBinomial(\mu_i, \phi_i)$, i is the index for region i

μ_i: expected count for region r[i]

ϕ_i: overdispersion parameter for region r[i]

$\beta^{cos}_{ri} \sim N(0, 2)$

2. Global covariate coefficients:
   $\theta_{depth} \sim N(0, 2)$
   $\theta_{lat} \sim N(0, 2)$
   $\theta_{lon} \sim N(0, 2)$
   $\theta_{lagmag} \sim N(0, 2)$
3. Dispersion model:
   $\gamma_1 \sim N(0, 0.5)$
   $\gamma_2 \sim N(0, 0.5)$
   $\gamma_3 \sim N(0, 0.5)$
   $\phi_{base} \sim Exponential(1)$

2. Mean function μ_i

$$log(\mu_i) = \alpha_{ri} + \beta^{time}_{ri} * time_{ri} + \beta^{sin}_{ri} * sin(\frac{2\pi * time_{ri}}{12}) + \beta^{cos}_{ri} * cos(\frac{2\pi * time_{ri}}{12}) + \theta_{depth} * depth_{ri} + \theta_{lat} * lat_{ri} \theta_{lon} * lon_{ri} + \theta_{lagmag} * lagmag_{ri}$$

3. Dispersion function ϕ_i

$$log(\phi_i) = \gamma_1 * Nst_{ri} + \gamma_2 * RMS_{ri} + \gamma_3 * CLO_{ri} + log(\phi_{base})$$

5.3 Prior choices, parameter interpretations and more explanations of the model

α_{ri}: This region-specific intercept controls baseline log-rate of earthquakes for each region. We chose a normal prior centered at 0 because we assume most regions start around the global average log-count, but can deviate. It encourages moderate variation across regions while discouraging extreme baselines unless supported by data.

β^{time}_{ri}, β^{sin}_{ri}, β^{cos}_{ri}: These allow each region to have independent trends and seasonality. In the model, $time_i$ is trying to capture long-term trends in earthquake activity since many natural phenomena could evolve slowly over time (Bahdanau, Cho, & Bengio, 2014). The sin and cos are trying to capture seasonal patterns in earthquake frequency since many earthquake risks can be affected by annual cycles like: seasonal groundwater loading, temperature changes and seasonal stress changes in faults (Hsu et al., 2021).

θ_{depth}, θ_{lat}, θ_{lon}, θ_{lagmag}: These are shared effects of depth, location, and prior magnitudes on expected counts. We include depth in the model since some literature indicates that depth is related to the magnitude of earthquakes (Scholz, 1998). We include lagmag since the simpler model above indicates that prior magnitudes have some indications of future magnitudes.

For the dispersion model, we used Normal(0, 0.5) priors on the coefficients of Nst, RMS, and CLO to ensure stable estimation. These predictors were chosen because they reflect measurement quality, allowing the model to adjust variance based on the reliability of each observation.

5.4 Posterior Dignostics and Predictive Checks

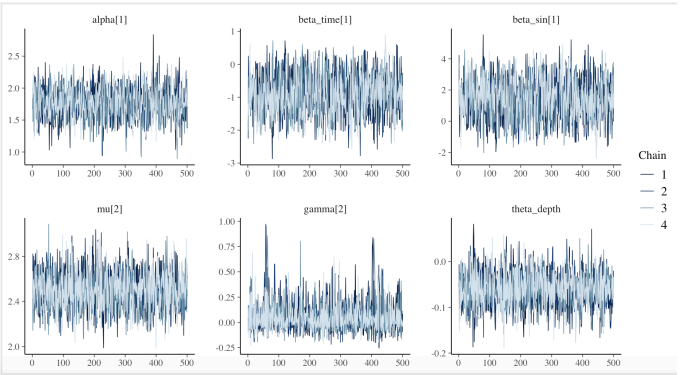

Figure 5.1: trace plots for some key parameters.

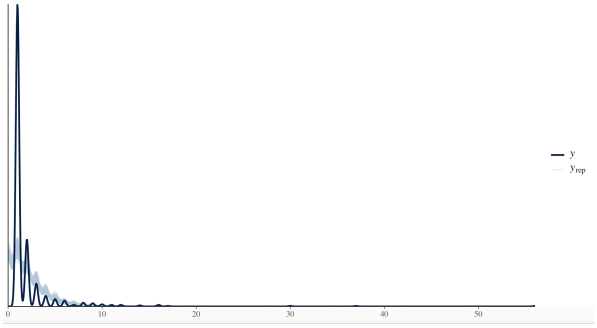

Figure 5.2: Posterior Predictive Check density overlay plot.

For this model, all chains mixed well as seen in the trace plots (Fig 5.1), and all Rhat values are below 1.01. The effective sample size is high, with most parameters exceeding 1500 and many above 2000, indicating reliable estimation and good convergence. There were no divergent transitions or tree-depth saturations reported during sampling. The posterior predictive check (Fig 5.2) shows that the model captures the general distribution of observed counts, although some underestimation in the lower counts events particularly the spikes at count = 1. The model provides a better fit to higher count values. This pattern suggests that the model sacrifices some accuracy in modeling very frequent, low-count observations in order to better capture the tail behavior. Since higher counts are more informative, the model's improved performance in this region is especially valuable for the application.

5.5 Forecasting and Out-of-Sample Evaluation

	RMSE	MAE
NegBin model	3.8	1.43

Table 5.1: Metrics for evaluating the prediction result of the NegBin model

To further test the performance of our model, we randomly select 20% of the data as a testing set. The model's pointwise 95% predictive credible intervals cover 96% of test observations, reflecting adequate calibration. Evaluation metrics on the test set yielded an RMSE of 3.8 and MAE of 1.43 (Table 5.1), which demonstrate predictive performance better than the simpler model. The model may still be slightly overconservative in the tails but provides improved mixing and robustness across predictors.

6. Model Comparison, Discussion and Conclusion

We evaluated two Bayesian models for forecasting monthly counts of magnitude ≥4 earthquakes in Northern California: a Poisson-based state-space model and a more advanced hierarchical Negative Binomial (NegBin) model. The Poisson model offers simplicity and captures overall trends well, but struggles with overdispersion and extreme values, often underestimating large counts and overestimating smaller ones. Its predictive intervals tend to be overly wide, with 99.8% test coverage. Another downside of this model is that it only has short-term memory since it only considers one previous time step, causing inaccurate long-term forecasts.

The Negative Binomial model addresses these issues by incorporating spatial hierarchies, seasonal patterns, and a covariate-dependent dispersion structure. It achieved improved predictive accuracy (RMSE = 3.8 vs. 4.67; MAE = 1.43 vs. 2.79), better calibration (96% coverage), and showed strong MCMC diagnostics with effective sample sizes frequently exceeding 2000 and all Rhat values < 1.01. Trace plots and posterior predictive checks confirmed robust convergence and improved tail behavior.

Despite these strengths, a key limitation remains: accurately modeling extreme high-count events is inherently difficult due to their rarity. While our model improves fit in the tail, some underestimation persists. Future work could explore explicit mixture models, heavy-tailed priors, or data augmentation techniques to better capture rare but impactful seismic activity.

Overall, the hierarchical Negative Binomial model provides more accurate, interpretable, and reliable predictions, making it better suited for real-world applications in seismic forecasting.

References (APA)

Attribution: Waveform data, metadata, or data products for this study were accessed through the Northern California Earthquake Data Center (NCEDC), doi:10.7932/NCEDC.

Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper).

Kagan, Y. Y. (2010). Statistical distributions of earthquake numbers: consequence of branching process. Geophysical Journal International, 180(3), 1313-1328.

Natvig, B., & Tvete, I. F. (2004). A comparison of Bayesian hierarchical space-time models for earthquake data. Preprint series. Statistical Research Report http://urn. nb. no/URN: NBN: no-23420.

Natvig, B., & Tvete, I. F. (2007). Bayesian hierarchical space–time modeling of earthquake data. Methodology and Computing in Applied Probability, 9, 89-114.

NCEDC (2014), Northern California Earthquake Data Center. UC Berkeley Seismological Laboratory. Dataset. doi:10.7932/NCEDC.

Posterior analysis. Stan Docs. (n.d.). https://mc-stan.org/docs/reference-manual/analysis.html

Scholz, C. H. (1998). Earthquakes and friction laws. Journal of Geophysical Research: Solid Earth, 103(B10), 23983–23994. https://doi.org/10.1029/97JB01356

Waldhauser, F., & Schaff, D. P. (2008). Large-scale relocation of two decades of Northern California seismicity using cross-correlation and double-difference methods. Journal of Geophysical Research: Solid Earth, 113(B8).

Waldhauser, F. (2009). Near-real-time double-difference event location using long-term seismic archives, with application to Northern California. Bulletin of the Seismological Society of America, 99(5), 2736-2748.

Appendices

All codes are available on https://github.com/GerGerGai/Bayesian_TimeSeries_StateSpace_Modeling , here is a copy of some important code:

1. Stan Code for Poisson Model

```
data {
  int<lower=1> N;           // number of observations
  int<lower=0> count[N];    // response variable

  vector[N] depth;
  vector[N] lag_mag;

  int<lower=1> H;           // forecast horizon
  vector[H] depth_future;
  vector[H] lag_mag_future;
}

parameters {
  real alpha;               // intercept
  real<lower=-1,upper=1> phi_raw;
  real<lower=0> sigma;      // RW sd
  vector[N] z;              // non-centred states
  real theta_depth;
  real theta_lagmag;

}

transformed parameters {
  real phi = phi_raw;       // already constrained
  vector[N] log_lambda;
  log_lambda[1] = alpha + z[1]*sigma;
  for (t in 2:N)
    log_lambda[t] = alpha + phi*(log_lambda[t-1]-alpha) + z[t]*sigma;
}

model {
  // Priors
  alpha        ~ normal(0, 1);
  phi_raw      ~ uniform(-1, 1);
  sigma        ~ cauchy(0, 1);     // half-Cauchy automatically by <0>
  theta_depth   ~ normal(0, 0.5);
  theta_lagmag  ~ normal(0, 0.5);
  z            ~ normal(0, 1);


  // Likelihood
  for (t in 1:N)
    count[t] ~ poisson_log(log_lambda[t] +
                      theta_depth  * depth[t] +
                      theta_lagmag * lag_mag[t]);
}
```

```
generated quantities {
  //for posterior predictive checking
  int y_rep[N];

   for (t in 1:N)
    y_rep[t] = poisson_log_rng(log_lambda[t] +
                               theta_depth  * depth[t] +
                               theta_lagmag * lag_mag[t]);

  //for forecasting
  vector[H] log_lambda_fore;     // latent states
  int y_fore[H];                 // predictions
  real logl_prev = log_lambda[N];

  for (h in 1:H) {
    logl_prev = alpha + phi * (logl_prev - alpha)
                      + sigma * normal_rng(0, 1);
    log_lambda_fore[h] = logl_prev;
    y_fore[h] = poisson_log_rng(
                  logl_prev
                  + theta_depth  * depth_future[h]
                  + theta_lagmag * lag_mag_future[h]);
  }

}
```

2. R Code for Poisson Model

```
library(dplyr)
library(lubridate)
library(rstan)
library(tidyr)
library(bayesplot)
library(scoringRules)
library(posterior)
library(ggplot2)
#install.packages("V8")


## Data Preparation
df <- read.csv("cleaned_earthquake_data.csv")
df$datetime <- as.POSIXct(paste(df$Date, df$Time), format="%Y/%m/%d %H:%M:%OS", tz="UTC") # combine
dates and times


df <- df %>%
  filter(!is.na(datetime))

df <- df %>%
  mutate(LagMag = lag(Mag)) %>%
```

```r
  ungroup()

agg_df <- df %>%
  mutate(month = floor_date(datetime, "month")) %>%
  group_by(month) %>%
  summarise(
    Count = sum(Mag >= 4, na.rm = TRUE),
    Depth = mean(Depth, na.rm = TRUE),
    LagMag = mean(LagMag, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  drop_na()


agg_df <- agg_df %>%
  mutate(across(c(Depth, LagMag), scale)) %>%
  mutate(across(everything(), ~ifelse(is.finite(.), ., 1e-3)))




################################################################################
####### SECTION: model 1: Poisson model ######################

H = 12 # forecast horizon
N_total = nrow(agg_df)
N_train = N_total - H


stan_data_1 <- list(
  N = N_train,
  count = agg_df$Count[1:N_train],
  depth = as.vector(agg_df$Depth)[1:N_train],
  lag_mag = as.vector(agg_df$LagMag)[1:N_train],

  H = H,
  depth_future = as.vector(agg_df$Depth)[(N_train+1):N_total],
  lag_mag_future = as.vector(agg_df$Depth)[(N_train+1):N_total]
)


init_fn_1 <- function() {
  R <- stan_data_1$R

  list(
    alpha = 0,
    phi_raw = 0,
    sigma = 0,
    mu_z = 0,
    sigma_z = 1,
    theta_depth = 0,
    theta_lagmag = 0
```

```
  )
}
set.seed(447)

fit1 <- stan(
  file = "RScripts/Poisson_simple.stan",
  data = stan_data_1,
  iter = 2000,
  chains = 4,
  seed = 42,
  control = list(adapt_delta = 0.8, max_treedepth = 10),
  init = "random"
)

########  model1 posterior check:
fit1

# 0. fast slow mixing:
mcmc_trace(fit1, pars = "alpha")
mcmc_trace(fit1, pars = "phi_raw")
mcmc_trace(fit1, pars = "sigma")
mcmc_trace(fit1, pars = c("alpha", "phi_raw", "sigma"),
           facet_args = list(nrow = 1))


# 1. convergence: Rhat and neff
# aim: Rhat<1.01, >400
summ = summary(fit1)$summary
summ
bad_rhat = sum(summ[,"Rhat"] > 1.01)
bad_neff  =  sum(summ[,"n_eff"] < 400)

ratios1 <- neff_ratio(fit1)
print(ratios1)

cat("No. of bad Rhat:", bad_rhat, "\n")
cat("No. of bad n_eff:", bad_neff, "\n")

# 2. sampler dignostic: divergent transitions and tree depth
# aim: 0 div, 0 hit max depth

sampler_pars = rstan::get_sampler_params(fit1, inc_warmup = FALSE)
sampler_mat  = do.call(rbind, sampler_pars)

div_total = sum(sampler_mat[, "divergent__"])
cat("Total divergences:", div_total, "\n")

max_td = max(sampler_mat[, "treedepth__"])
cat("Maximum treedepth reached:", max_td, "\n")

# 3. autocorrelation
# aim: no long term autocorr,
```

```r
draws_arr <- as.array(fit1)
bayesplot::mcmc_acf(draws_arr, pars = "alpha", lags = 30)


# 4. posterior-predictive fit
posterior <- rstan::extract(fit1)
y_rep <- posterior$y_rep
ppc_dens_overlay(y = stan_data_1$count, yrep = y_rep[sample(4000, 200), ])
stan_data_1$count


## For discrete data ➡ better: rootogram
ppc_rootogram(
  y    = stan_data_1$count,
  yrep = y_rep[sample(4000, 50), ]  # 50 draws is usually enough
)

ppc_stat(y = stan_data_1$count, yrep = y_rep, stat = "mean")



#7. credible interval
# too conservative?
y_pred_ci <- apply(y_rep, 2, quantile, probs = c(0.025, 0.975))
mean(stan_data_1$count >= y_pred_ci[1, ] & stan_data_1$count <= y_pred_ci[2, ])



########  model1 forecasting check:
y_test = agg_df$Count[(N_train + 1):N_total]
y_test

fc_pois <- as_draws_matrix(rstan::extract(fit1, pars = "y_fore")$y_fore)

rmse <- function(mat, obs) {
  sqrt( mean( (colMeans(mat) - obs)^2 ) )
}
mae  <- function(mat, obs) {
  mean( abs(colMeans(mat) - obs) )
}

metrics <- tibble(
  model = c("Poisson-SSM"),
  RMSE  = c(rmse(fc_pois, y_test)), #stub for model2's prediction
  MAE   = c(mae(fc_pois, y_test))
)


logs_pois <- mean(logs_sample(y_test, t(fc_pois)))
logs_nb   <- 1

metrics <- metrics |>
  mutate(
```

```
    LogS = c(logs_pois)
  )

print(metrics)

pi95_pois <- mean(
  y_test >= apply(fc_pois, 2, quantile, 0.025) &
    y_test <= apply(fc_pois, 2, quantile, 0.975)
)
pi95_pois

y_test
apply(fc_pois, 2, quantile, c(0.025, 0.5,  0.975))

ci_limits_fore = apply(fc_pois, 2, quantile, c(0.025, 0.975))
ci_limits_fore[1,]


inside_ci =  y_test >= ci_limits_fore[1,] & y_test <= ci_limits_fore[2,]

tmpdf = data.frame(
  x = 1:H,
  y = y_test,
  ymin = ci_limits_fore[1, ],
  ymax = ci_limits_fore[2, ],
  inside_ci = inside_ci
)

#code referencing ex8's plot
ggplot(tmpdf, aes(x = x, y = y,ymin = ymin, ymax = ymax,color = inside_ci)) +
  geom_point() +
  geom_errorbar() +
  theme_minimal() +
  scale_x_continuous(
    breaks = 1:12,
    labels = 1:12
  ) +
  ggtitle("Observed Counts, and 95% Prediction Credible Intervals (Poisson Model)") +
  labs(x = "h", y = "Earthquake Count")
```

3. Stan Code for NegBin Model

```
data {
  int<lower=1> N;
  int<lower=1> R;
  int<lower=1, upper=R> region[N];
```

```
  int<lower=0> count[N];
  vector[N] time;
  vector[N] depth;
  vector[N] lat;
  vector[N] lon;
  vector[N] lag_mag;
  vector[N] nst;
  vector[N] rms;
  vector[N] clo;
}

parameters {
  vector[R] alpha;
  vector[R] beta_time;
  vector[R] beta_sin;
  vector[R] beta_cos;

  real theta_depth;
  real theta_lat;
  real theta_lon;
  real theta_lagmag;

  vector[3] gamma;
  real<lower=0> phi_base;
}

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

  for (i in 1:N) {
    real season_sin = sin(2 * pi() * time[i] / 12);
    real season_cos = cos(2 * pi() * time[i] / 12);

    mu[i] = exp(
      alpha[region[i]] +
      beta_time[region[i]] * time[i] +
      beta_sin[region[i]] * season_sin +
      beta_cos[region[i]] * season_cos +
      theta_depth * depth[i] +
      theta_lat   * lat[i] +
      theta_lon   * lon[i] +
      theta_lagmag * lag_mag[i]
    );

    real log_phi = gamma[1] * nst[i] + gamma[2] * rms[i] + gamma[3] * clo[i];
    phi[i] = exp(log_phi) * phi_base;
  }
}

model {
  alpha ~ normal(0, 2);
  beta_time ~ normal(0, 2);
```

```
  beta_sin ~ normal(0, 2);
  beta_cos ~ normal(0, 2);

  theta_depth ~ normal(0, 2);
  theta_lat   ~ normal(0, 2);
  theta_lon   ~ normal(0, 2);
  theta_lagmag ~ normal(0, 2);

  gamma ~ normal(0, 0.5);
  phi_base ~ exponential(1);

  for (i in 1:N)
    count[i] ~ neg_binomial_2(mu[i], phi[i]);
}

generated quantities {
  vector[N] y_rep;
  for (i in 1:N) {
    y_rep[i] = neg_binomial_2_rng(mu[i], phi[i]);
  }
}
```

4. R Code for NegBin Model

```
## Data Preparation
df <- read.csv("cleaned_earthquake_data.csv")
df$datetime <- as.POSIXct(paste(df$Date, df$Time), format="%Y/%m/%d %H:%M:%OS", tz="UTC") # combine
dates and times


df <- df %>%
  filter(!is.na(datetime) & Mag >= 4)

region_size <- 2
df$region_lat <- floor(df$Lat / region_size)
df$region_lon <- floor(df$Lon / region_size)
df$region <- paste0("Lat", df$region_lat, "_Lon", df$region_lon)
df$region_idx <- as.numeric(as.factor(df$region))

# Time index as continuous covariate (in months since first quake)
origin_time <- min(df$datetime)
df$TimeIndex <- as.numeric(difftime(df$datetime, origin_time, units = "days")) / 30

# Add sine/cosine seasonal terms for monthly seasonality
df$SeasonSin <- sin(2 * pi * df$TimeIndex / 12)
df$SeasonCos <- cos(2 * pi * df$TimeIndex / 12)

df <- df %>%
  arrange(region, TimeIndex) %>%
  group_by(region) %>%
```

```
  mutate(LagMag = lag(Mag)) %>%
  ungroup()


agg_df <- df %>%
  mutate(month = floor_date(datetime, "month")) %>%
  group_by(region, month) %>%
  summarise(
    Count = n(),
    Depth = mean(Depth, na.rm = TRUE),
    Lat = mean(Lat, na.rm = TRUE),
    Lon = mean(Lon, na.rm = TRUE),
    LagMag = mean(LagMag, na.rm = TRUE),
    TimeIndex = mean(TimeIndex, na.rm = TRUE),
    SeasonSin = mean(SeasonSin, na.rm = TRUE),
    SeasonCos = mean(SeasonCos, na.rm = TRUE),
    Nst = mean(Nst, na.rm = TRUE),
    RMS = mean(RMS, na.rm = TRUE),
    Clo = mean(Clo, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  drop_na()

agg_df <- agg_df %>%
  mutate(across(c(Depth, Lat, Lon, LagMag, TimeIndex, SeasonSin, SeasonCos, Nst, RMS, Clo), scale))
%>%
  mutate(across(everything(), ~ifelse(is.finite(.), ., 1e-3)))

set.seed(42)
n <- nrow(agg_df)
train_idx <- sample(1:n, size = floor(0.8 * n), replace = FALSE)
agg_train <- agg_df[train_idx, ]
agg_test <- agg_df[-train_idx, ]

stan_data <- list(
  N = nrow(agg_train),
  R = length(unique(df$region)),
  region = as.integer(factor(agg_train$region)),
  count = agg_train$Count,
  time = as.vector(agg_train$TimeIndex),
  depth = as.vector(agg_train$Depth),
  lat = as.vector(agg_train$Lat),
  lon = as.vector(agg_train$Lon),
  lag_mag = as.vector(agg_train$LagMag),
  nst = as.vector(agg_train$Nst),
  rms = as.vector(agg_train$RMS),
  clo = as.vector(agg_train$Clo)
)

init_fn <- function() {
  R <- stan_data$R

  list(
```

```
    alpha = rnorm(R, 0, 0.1),
    beta_time = rnorm(R, 0, 0.1),
    beta_sin = rnorm(R, 0, 0.1),
    beta_cos = rnorm(R, 0, 0.1),

    theta_depth = 0,
    theta_lat = 0,
    theta_lon = 0,
    theta_lagmag = 0,

    gamma = rep(0, 3),

    mu_alpha = 0,
    sigma_alpha = 1,

    mu_beta_time = 0,
    sigma_beta_time = 1,

    mu_beta_sin = 0,
    sigma_beta_sin = 1,

    mu_beta_cos = 0,
    sigma_beta_cos = 1
  )
}

options(mc.cores = 4)

fit <- stan(
  file = "RScripts/NegBin.stan",
  data = stan_data,
  iter = 1000,
  chains = 4,
  seed = 42,
  control = list(adapt_delta = 0.99, max_treedepth = 15),
  init = "random"
)

# save the model to disk first:
saveRDS(fit, file = "trained_model_tail_improve2_good?.rds")
# to read: fit <- readRDS("trained_model.rds")

# check mixing/convergence:
print(fit)
summary(fit)$summary

# check fitness of the model: the result seems good, though not perfect
posterior <- rstan::extract(fit)
y_rep <- posterior$y_rep
ppc_dens_overlay(y = stan_data$count, yrep = y_rep[1:300, ])

# trace plots for main parameters
posterior_array <- as.array(fit)
```

```r
mcmc_trace(posterior_array, pars = c(
  "alpha[1]",
  "beta_time[1]",
  "beta_sin[1]",
  "mu[2]",
  "gamma[2]",
  "theta_depth"
))

y_pred_ci <- apply(y_rep, 2, quantile, probs = c(0.05, 0.95))
mean(stan_data$count >= y_pred_ci[1, ] & stan_data$count <= y_pred_ci[2, ])

y_pred_mean <- colMeans(y_rep)
plot(y_pred_mean, stan_data$count,
     xlab = "Predicted mean", ylab = "Observed count", main = "Fit scatterplot")
abline(0, 1, col = "red")

resid <- stan_data$count - y_pred_mean
plot(resid, main = "Residuals", ylab = "Observed - Predicted")
abline(h = 0, col = "red", lty = 2)

region_test <- as.integer(factor(agg_test$region))
time_test   <- agg_test$TimeIndex
depth_test  <- agg_test$Depth
lat_test    <- agg_test$Lat
lon_test    <- agg_test$Lon
lagmag_test <- agg_test$LagMag
nst_test    <- agg_test$Nst
rms_test    <- agg_test$RMS
clo_test    <- agg_test$Clo

n_test <- nrow(agg_test)
n_draws <- length(posterior$theta_depth)
y_rep_test <- matrix(NA, nrow = n_draws, ncol = n_test)

for (d in 1:n_draws) {
  eta <- posterior$alpha[d, region_test] +
    posterior$beta_time[d, region_test] * time_test +
    posterior$beta_sin[d, region_test] * sin(2 * pi * time_test / 12) +
    posterior$beta_cos[d, region_test] * cos(2 * pi * time_test / 12) +
    posterior$theta_depth[d] * depth_test +
    posterior$theta_lat[d]   * lat_test +
    posterior$theta_lon[d]   * lon_test +
    posterior$theta_lagmag[d] * lagmag_test

  mu <- exp(eta)

  phi <- exp(
    posterior$gamma[d, 1] * nst_test +
      posterior$gamma[d, 2] * rms_test +
      posterior$gamma[d, 3] * clo_test
  ) * posterior$phi_base[d]
```

```
  y_rep_test[d, ] <- rnbinom(n_test, size = phi, mu = mu)
}

y_pred_mean <- colMeans(y_rep_test)

plot(y_pred_mean, agg_test$Count,
     xlab = "Predicted Mean", ylab = "Observed Count",
     main = "Test Set Prediction")
abline(0, 1, col = "red")

rmse <- sqrt(mean((y_pred_mean - agg_test$Count)^2))
y_lower <- apply(y_rep_test, 2, quantile, probs = 0.025)
y_upper <- apply(y_rep_test, 2, quantile, probs = 0.975)
coverage <- mean(agg_test$Count >= y_lower & agg_test$Count <= y_upper)

cat("Test RMSE:", round(rmse, 2), "\n")
cat("95% Coverage:", round(coverage * 100, 2), "%\n")

mae <- mean(abs(y_pred_mean - agg_test$Count))
logS <- mean(logs_sample(agg_test$Count, t(y_rep_test)))

cat("Test MAE:", round(mae, 2), "\n")
cat("Test logS:", round(logS, 3), "\n")
```

5. For some Exploratory Data Analysis of the dataset, please refer to the EDA.R file under folder RScripts in our github repo.