Bayesian Hierarchical Modelling For Tornadoes

Motivating Examples Via Metaculus

Author: 118777

Q2 2024

1 Question 22307

The resolution criteria for Metaculus question 22307, which is part of the Bridgewater Forecasting Contest, reads:

> This question will resolve as the number of tornadoes in the United States in April of 2024, according to the Storm Prediction Center's preliminary tornado summary.
>
> If the resolution value is below the lower bound or above the upper bound, the question resolves outside the respective bound.
>
> The question will resolve as the number of tornadoes shown for the month of April 2024 when accessed by Metaculus on May 3, 2024. If there is a discrepancy between the "Map" and "Tables" views, the question resolves to the figure displayed on the "Map" view.

The National Oceanic and Atmospheric Administration's (NOAA) Storm Prediction Center permits data downloading (though it seems quite tedious), so computional modelling is possible. The data available for us to use includes tornadoes, wind, and hail counts for each state, by month and year (1950 to 2024). For more information about data collection, see the Appendix.

So, how can we model the expected number of tornadoes for April 2024 in the United States?

2 Model A

Well, the number of tornadoes in state s, month m, and year t can be described as

$$Y_{smt} = \alpha_s + \gamma_m + \delta_t + \theta_{mt}$$

(a multilevel model) where α_s is the random intercept which describes the unique tornado activity of each state s, γ_m is the fixed effect describing seasonal variation across each month m, δ_t is the fixed effect describing trends in tornado activity across each year t, and θ_{mt} is an interaction term for describing how the effect of a given month might change over the years.

Since we are interested in the *rate* of tornadoes, we believe $Y_{smt} \sim \text{Poisson}(\lambda_{smt})$, where $\log(\lambda_{smt}) = \alpha_s + \gamma_m + \delta_t + \theta_{mt}$. Let us assume

$$\sigma_\alpha, \sigma_\gamma, \sigma_\delta, \sigma_\theta \sim \text{Exponential}(1.0)$$
$$\alpha_s \sim \mathcal{N}(0, \sigma_\alpha)$$
$$\gamma_m \sim \mathcal{N}(0, \sigma_\gamma)$$
$$\delta_t \sim \mathcal{N}(0, \sigma_\delta)$$
$$\theta_{mt} \sim \mathcal{N}(0, \sigma_\theta)$$

As for predicting the number of tornadoes during April 2024, across all states (please forgive the poor form present in the `posterior_predictive_distribution` function and the lack of docstrings—the author will get to these in time):

Listing 1: Model Implementation In Numpyro

```
def tornado_modelA(state=None, month=None, year=None,
   tornadoes=None):
    num_states = len(np.unique(state))
    num_months = len(np.unique(month))
    num_years = len(np.unique(year))

    sigma_alpha = npro.sample("sigma_alpha", dist.Exponential(1.0))
    sigma_gamma = npro.sample("sigma_gamma", dist.Exponential(1.0))
    sigma_delta = npro.sample("sigma_delta", dist.Exponential(1.0))
    sigma_theta = npro.sample("sigma_theta", dist.Exponential(1.0))

    alpha = npro.sample(
        "alpha", dist.Normal(0, sigma_alpha),
            sample_shape=(num_states,)
    )
    gamma = npro.sample(
        "gamma", dist.Normal(0, sigma_gamma),
            sample_shape=(num_months,)
    )
    delta = npro.sample(
        "delta", dist.Normal(0, sigma_delta),
            sample_shape=(num_years,)
    )
    theta = npro.sample(
        "theta",
        dist.Normal(0, sigma_theta),
        sample_shape=(num_months, num_years),
```

```python
    )

    lambda_ = jnp.exp(
        alpha[state] + gamma[month] + delta[year] + theta[month,
            year]
    )
    npro.sample("obs", dist.Poisson(lambda_), obs=tornadoes)


def inference(model: Callable, cf: dict, data: pl.DataFrame,
    save_path: str):
    nuts_kernel = npro.infer.NUTS(
        model,
        dense_mass=True,
        max_tree_depth=cf["inference"]["max_tree_depth"],
        init_strategy=npro.infer.init_to_median,
    )
    mcmc = npro.infer.MCMC(
        nuts_kernel,
        num_warmup=cf["inference"]["num_warmup"],
        num_samples=cf["inference"]["num_samples"],
        num_chains=cf["inference"]["num_chains"],
        progress_bar=cf["inference"]["progress_bar"],
    )
    rng_key = jr.PRNGKey(cf["reproducibility"]["seed"])
    mcmc.run(
        rng_key,
        state=jnp.array(data["State"].to_numpy()),
        month=jnp.array(data["Month"].to_numpy()),
        year=jnp.array(data["Year"].to_numpy()),
        tornadoes=jnp.array(data["Tornado"].to_numpy()),
    )
    if cf["inference"]["summary"]:
        mcmc.print_summary()
    return mcmc.get_samples()

def posterior_predictive_distribution(
    samples, model, cf
):
    predictive = npro.infer.Predictive(
        model, posterior_samples=samples
    )
    rng_key = jr.PRNGKey(cf["reproducibility"]["seed"])
```

```
    post_pred = predictive(
        rng_key,
        state=jnp.array(list(range(50))),
        month=jnp.array([3]),
        year=jnp.array([2024])
    )
    return post_pred
```

2.1 Adding Wind As A Predictor

3 Appendix

3.1 Random Intercepts And Slopes

From Wikipedia, accessed 2024-04-20.

> Before conducting a multilevel model analysis, a researcher must decide on several aspects, including which predictors are to be included in the analysis, if any. Second, the researcher must decide whether parameter values (i.e., the elements that will be estimated) will be fixed or random.[2][5][4] Fixed parameters are composed of a constant over all the groups, whereas a random parameter has a different value for each of the groups.[4] Additionally, the researcher must decide whether to employ a maximum likelihood estimation or a restricted maximum likelihood estimation type.[2]
>
> **Random intercepts model**
>
> A random intercepts model is a model in which intercepts are allowed to vary, and therefore, the scores on the dependent variable for each individual observation are predicted by the intercept that varies across groups.[5][8][4] This model assumes that slopes are fixed (the same across different contexts). In addition, this model provides information about intraclass correlations, which are helpful in determining whether multilevel models are required in the first place.[2]

Random slopes model

A random slopes model is a model in which slopes are allowed to vary according to a correlation matrix, and therefore, the slopes are different across grouping variable such as time or individuals. This model assumes that intercepts are fixed (the same across different contexts).[5]

Random intercepts and slopes model

A model that includes both random intercepts and random slopes is likely the most realistic type of model, although it is also the most complex. In this model, both intercepts and slopes are allowed to vary across groups, meaning that they are different in different contexts.[5]

3.2 Gathering Data From NOAA

Note that there are preliminary and final counts. Since some entries do not have final counts, even though many years have passed, and since the question resolves based on the preliminary counts,

3.3 Full Code