Bayesian Beta Regression Model for Coral Bleaching

1. Model Specification

To model the proportion of coral bleaching (mu_i), we use a Beta regression with a logit link function:

logit(mu_i) = f(Date_Year_i, City_Town_Name_i) + g(Lat_i, Lon_i) + X_i * beta

where:

- f(Date_Year_i, City_Town_Name_i) is a Gaussian Process (GP) over time, modeling non-linear temporal trends within each city.
- g(Lat_i, Lon_i) is a tensor product spline (t2(Lat, Lon)), capturing continuous spatial variation.
- X_i * beta represents environmental predictors (e.g., Turbidity, Depth, Cyclone Frequency).
- The Beta distribution is used to account for the bounded nature of bleaching proportions:

  y_i ~ Beta(mu_i * phi, (1 - mu_i) * phi)

where phi is the precision parameter.

2. Model Comparison

We tested multiple models to determine the best approach for capturing temporal and spatial variation in coral bleaching. The candidate models included:

- Linear models with Date_Year as a fixed effect.
- Random effects models, where City_Town_Name was treated as a random intercept.
- Gaussian Process (GP) models, both with and without city-specific trends.
- Spatial models, incorporating either Lat and Lon as fixed effects or a smooth spatial term (t2(Lat, Lon)).

The final model was selected using Leave-One-Out Cross-Validation (LOO-CV), ensuring it provided

the best balance between fit and complexity.

3. Model Comparison Results

Model	Temporal Structure	Spatial Structure	LOO Score
Model A	Linear (Date_Year)	None	-XXXX
Model B	Linear (Date_Year)	Lat/Lon as Fixed	-XXXX

Model C	Global GP	None	-XXXX
Model D	City-Specific GP	None	**Lowest LOO**
Final Model	City-Specific GP	Spatial Smoother (t2)	Slightly lower LOO than Mode

4. Justification of the Final Model

The best-fitting model included a Gaussian Process (GP) for temporal variation within each city, capturing nonlinear bleaching trends that vary by location. Linear models failed to capture these trends, and adding a spatial smoother (t2(Lat, Lon)) provided minimal improvement. Thus, the final model includes:

- gp(Date_Year, by = City_Town_Name) to model city-specific bleaching trends over time.
- t2(Lat, Lon) to account for spatial variation within each city.

This combination ensures that both temporal and spatial dependencies are accounted for.

5. How GP Affects the Mean

Gaussian Processes (GPs) do not impose a fixed mean shift like a standard regression term. Instead, they model deviations from the baseline mean in a flexible, data-driven way. In our model:

y_i = mu + f(Date_Year_i, City_Town_Name_i) + epsilon_i

where:

- mu represents the global mean (baseline trend).
- f(Date_Year_i, City_Town_Name_i) ~ GP(0, k(time)) models deviations from this mean.
- epsilon_i is residual noise.

GPs affect the mean **only where the data suggests strong deviations from a linear trend**. This allows for **nonlinear time variation** while preserving interpretable baseline effects.

Including both a fixed effect for Date_Year and gp(Date_Year) can allow for both global trends and flexible deviations.

6. Prior Distributions

To ensure regularization and prevent overfitting, we use weakly informative priors:

- Fixed-effect coefficients: beta ~ Normal(0, 5)
- GP variance parameter: sigma_f^2 ~ HalfCauchy(0, 2)
- GP length scale: length_scale ~ Gamma(2, 1)

- Spatial smoother coefficients: alpha_jk ~ Normal(0, 3)
- Beta precision parameter: phi ~ Gamma(0.1, 0.1)

7. Final Model Code in `brms`

```
formulaBleaching_beta <- bf(
  PercentBleachingBounded ~
    gp(Date_Year, by = City_Town_Name) +
    t2(Lat, Lon) +
    Distance_to_Shore + Exposure + Turbidity + Cyclone_Frequency +
    Depth_m + Windspeed + ClimSST + SSTA + TSA + TSA_DHW,
  family = Beta(link = "logit")
)

priors <- c(
  prior(normal(0, 5), class = "b"),
  prior(gamma(2, 1), class = "gp", coef = "Date_Year"),
  prior(half_cauchy(0, 2), class = "gp", coef = "Date_Year"),
  prior(normal(0, 3), class = "sds", coef = "t2LatLon"),
  prior(gamma(0.1, 0.1), class = "phi")
)

betaFit <- brm(
  formulaBleaching_beta,
  data = procData2,
  prior = priors,
  chains = 4, iter = 2000, warmup = 1000,
  control = list(adapt_delta = 0.95)
)
```