Project Report

Contents

1 Literature Review

The use of machine learning (ML) in spatial environmental modeling has grown rapidly in recent years, with Random Forests (RF) being a particularly popular choice due to their ability to capture nonlinearities and interactions in complex datasets (**??**). However, standard RF assumes independent and identically distributed (i.i.d.) data, which is rarely true for spatial datasets where spatial autocorrelation is present (**?**). This assumption leads to biased estimates and overconfident predictions if spatial dependence is not addressed. In response, researchers have developed several strategies to adapt RF for spatial applications. Spatial extensions, such as RFsp and RF residual kriging (RF-RK), attempt to incorporate spatial information either through additional covariates or post-processing methods, though these approaches often lack a solid theoretical foundation (**?**). Saha et al. recently introduced RF-GLS, a novel framework that integrates spatial dependence directly into the RF learning process using generalized least squares (GLS), demonstrating both improved predictive performance and theoretical consistency under spatial correlation structures, particularly those modeled by Matérn Gaussian processes. In parallel, spatial validation techniques have also evolved to account for autocorrelation in model evaluation. Standard cross-validation tends to overestimate model performance in spatial contexts, prompting the development of spatially-aware approaches such as spatial blocking (**??**). From a statistical modeling perspective, hierarchical models and spatial mixed-effects frameworks provide flexible tools to model spatial structure and uncertainty explicitly (**?**), while additive models also allow for smooth, nonlinear effects of covariates in a spatial context (**?**). Variable selection is another key challenge in spatial ML applications; Meyer et al. (**?**) emphasized the need for selecting covariates that reflect spatial processes rather than just fitting patterns, a principle echoed in applications to environmental interpolation (**?**). In the context of coral bleaching, recent studies have highlighted the importance of capturing spatial and temporal dynamics to understand and predict bleaching events. Hughes et al. (**?**) documented widespread changes in coral reef assemblages due to global warming, while Hoegh-Guldberg et al. (**?**) reviewed the broader implications of climate change and ocean acidification on reef ecosystems. McClanahan et al. (**?**) explored the complexity of thermal exposure patterns during the 2016 El Niño and demonstrated that traditional indices such as Degree Heating Weeks (DHW) fail to fully capture bleaching risk, advocating for more nuanced predictors that consider duration, bimodality, and geography. Similarly, Ainsworth et al. (**?**) showed that prior exposure history and reef-specific factors significantly influence coral responses, suggesting a need to model local adaptation and historical context. Collectively, these studies emphasize that modeling coral bleaching requires not only accurate thermal predictors but also the integration of spatial dependencies in both modeling and validation stages. The recent development of spatially-explicit ML models like RF-GLS, combined with robust validation protocols, represents a promising direction for improving predictive accuracy and ecological relevance in coral reef monitoring and conservation.

2 Introduction

2.1 Spatially Dependent Data

Spatially dependent data refers to observations that are correlated based on their spatial proximity. This means that the value of a variable at one location is influenced by the values of the same variable at nearby locations. This type of data is common in fields like geography, ecology, and environmental science.

Spatially dependent data can be modeled using the spatial linear mixed model:

$$Y = X\beta + w(\ell) + \varepsilon$$

where Y is the response variable, X is a matrix of covariates, β is a vector of fixed effects coefficients, $w(\ell)$ is a spatial random effect at location ℓ, and ε is the error term.

For the spatial random effect $w(\ell)$, a Gaussian Process with a Matérn covariance function is often used:

$$\mathrm{Cov}(w(\ell_i), w(\ell_j)) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}d}{\rho}\right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}d}{\rho}\right)$$

where $d = ||\ell_i - \ell_j||$ is the distance between locations ℓ_i and ℓ_j, ν is the smoothness parameter, σ^2 is the variance, ρ is the range parameter, $\Gamma(\nu)$ is the gamma function, and K_ν is the modified Bessel function of the second kind.

2.2 Bootstrapping Methods

Bootstrapping is a resampling technique used to estimate the distribution of a statistic by resampling with replacement from the observed data. This method is widely used in statistical inference and machine learning to improve model robustness.

2.2.1 Standard Bootstrapping

Given a dataset $D = \{X_1, X_2, ..., X_n\}$ of size n, standard bootstrapping involves the following steps:

1. Draw n samples with replacement from D to create a bootstrap sample $D^* = \{X_1^*, X_2^*, ..., X_n^*\}$.
2. Compute the statistic of interest (e.g., mean, variance, regression coefficient) on D^*.
3. Repeat the above steps B times to obtain B bootstrap estimates.
4. Compute the empirical distribution of the bootstrap estimates to approximate the true distribution.

2.2.2 Bootstrapping in Random Forests

Random forests utilize bootstrapping to create diverse training datasets for each decision tree. The procedure is as follows:

1. Given a dataset D of size n, draw n samples with replacement to create a bootstrap sample D^*.
2. Train a decision tree on D^*.
3. Repeat the above process T times to construct an ensemble of T trees.
4. Aggregate the predictions of all trees (e.g., majority vote for classification, mean prediction for regression).

This method introduces randomness, reduces overfitting, and improves generalization.

2.2.3 GLS Bootstrapping Approach

The Generalized Least Squares (GLS) bootstrapping method accounts for spatial dependence in the data. Unlike standard bootstrapping, GLS bootstrapping incorporates the covariance structure estimated from the data to create more realistic resampling distributions.

Mathematical Formulation: Given the spatial regression model:

$$Y = X\beta + \varepsilon, \quad \text{where} \quad \varepsilon \sim \mathcal{N}(0, \Sigma)$$

where Σ is the covariance matrix capturing spatial dependence.

The GLS estimator is given by:

$$\hat{\beta}_{GLS} = (X^T \Sigma^{-1} X)^{-1} X^T \Sigma^{-1} Y$$

The GLS bootstrapping procedure consists of:

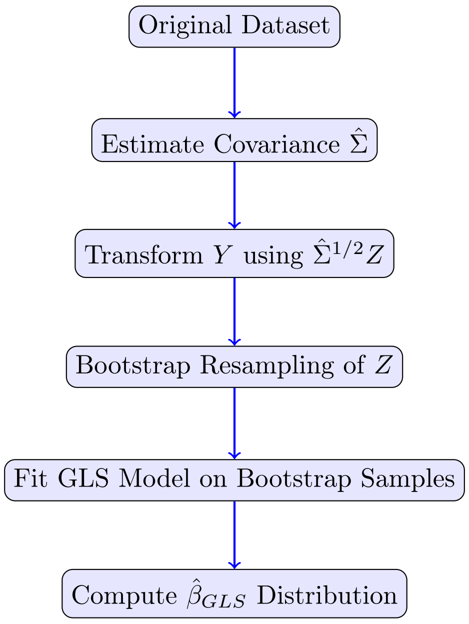
1. Estimate the covariance structure $\hat{\Sigma}$ from the residuals of an initial model fit.
2. Transform the response variable:

$$\hat{\Sigma}^{-1/2} Y = (\hat{\Sigma}^{-1/2} X)\beta + \hat{\Sigma}^{-1/2}\varepsilon, \quad \hat{\Sigma}^{-1/2}\varepsilon \sim \mathcal{N}(0, I)$$

$$Y^* = X^* \hat{\beta} + Z, \quad Z \sim \mathcal{N}(0, I)$$

3. Resample Z to generate multiple bootstrap samples.
4. Refit the model on each bootstrap sample and compute $\hat{\beta}_{GLS}$.
5. Aggregate bootstrap estimates to compute confidence intervals.

2.2.4 Diagrammatic Representation


Original Dataset
Estimate Covariance $\hat{\Sigma}$
Transform Y using $\hat{\Sigma}^{1/2} Z$
Bootstrap Resampling of Z
Fit GLS Model on Bootstrap Samples
Compute $\hat{\beta}_{GLS}$ Distribution

3 Understanding the Aim & the Dataset

3.1 Aim

The primary objective of this research is to investigate the factors influencing coral reef resilience in response to climate change and anthropogenic stressors. The study evaluates the effectiveness of different conservation strategies in mitigating the impacts of environmental changes on coral ecosystems. Specifically, it analyzes the patterns and drivers of coral bleaching events across multiple reef systems, assesses the role of local management interventions in enhancing reef recovery and resilience, investigates the interactions between climate-induced stress and human activities on coral reef health, and provides empirical data to support policy recommendations for sustainable coral reef conservation. By addressing these objectives, the study contributes to a deeper understanding of coral reef dynamics and informs future conservation efforts aimed at preserving marine biodiversity.

The dataset focus on coral bleaching patterns during the 2016 El Niño event across the Indo-Pacific region. This comprehensive dataset represents one of the most extensive coordinated efforts to document coral bleaching responses and the underlying environmental factors that influence bleaching severity.

3.2 Dataset Overview

The dataset contains information from 226 reef sites spanning 60° of latitude and 140° of longitude across the Indo-Pacific region, from East Africa to Fiji. Data was collected through standardized underwater surveys of 30 coral colonies during the peak of the 2016 El Niño/Southern Oscillation thermal anomaly. The surveys were conducted within 21 days of peak thermal anomalies to ensure timely assessment of bleaching impacts.

3.3 Key Variables in the Dataset

The dataset includes 28 environmental variables, categorized as follows:

Table 1: Summary of Variables Across Categories

Category	Variable	Description
Location	Location	Geographic region (e.g., Kolombangara, Kenya fringing)
	Site	Specific site name within the location
	X and Y Coordinates	Latitude and longitude coordinates
	Management	Conservation status (e.g., open, restricted, no-take)
	Habitat Reef Type	Reef classification (e.g., lagoon, flat, bank, channel, crest)
Bleaching	Bleach Intensity	Primary response variable measuring coral bleaching severity
Temperature	AvgDHW-90Days	Average Degree Heating Weeks over 90 days
	MaxDHW-90Days	Maximum Degree Heating Weeks over 90 days
	DipStatistic	A measure of temperature distribution
	BimodalityCoefficient	Indicates whether temperature distribution is bimodal
	BimodalityRatio	Ratio of two identified bimodality peaks
Temp. Events	NumHighEvents-90Days	Number of high-temperature events exceeding the 90th percentile
	HighSpellDuration-90Days	Average duration of high-temperature events
	MaxHighSpellDuration-90Days	Maximum duration of high-temperature events
	AvgHighSpellTemp-90Days	Average temperature during high-temperature events
	HighSpell-Variance	Variance of high-temperature events
	LowSpell-Duration-90Days	Average duration of low-temperature events
	DHD-90Days	Degree heating days
Community	CCA	A multivariate index of coral community composition

3.4 Data Collection Methodology

Researchers used standardized underwater survey methods to assess coral bleaching. For each site, they characterized sea surface temperatures (SSTs) in the 90 days before the survey, quantifying the number, duration, and magnitude of anomalous temperatures based on the 10th SST quantile ("cold spells") and 90th SST quantile ("hot spells").

4 Preliminary Analysis of Data

4.1 Introduction

According to the data, our target variable is **bleach intensity**

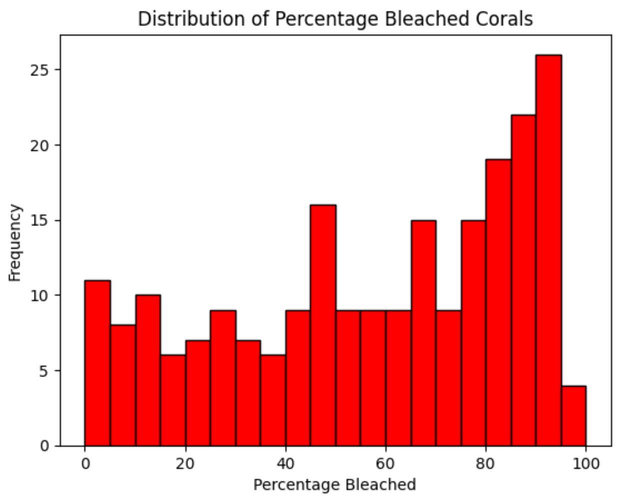


Figure 1: *Percentage of bleached coral vs frequency of locations*

These are the locations at which experiments for coral bleaching intensity were done

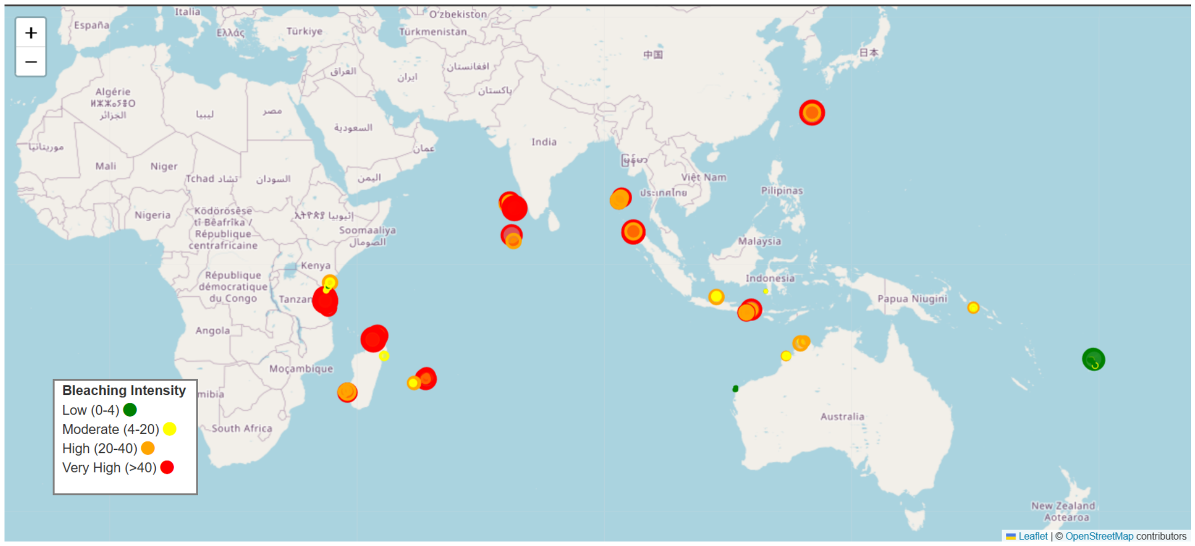


Figure 2:

4.2 Making ML Models to Predict Bleaching Intensity and Analyze Variable Influence

4.2.1 Variables (Covariates) Used in ML Model

`location, site, bleach_intensity, X, Y, management, habitat, average_dhw_90days, max_dhw_90days, dip_Statistic_sst, bimodality_coefficient, bimodality_ratio, n_h_spell_events_90Days, avg_high_spell_du max_high_spell_duration_90days, avg_high_spell_rise_90days, avg_spell_peak, sd_spell_peak, avg_low_spell_duration_90days, dhd_mmmplus1, CA1`

4.2.2 Using Simple Linear Regression

- **R^2 Score**: 0.5212550513577265
- **Observed vs Predicted Bleach Intensity on Test Data**

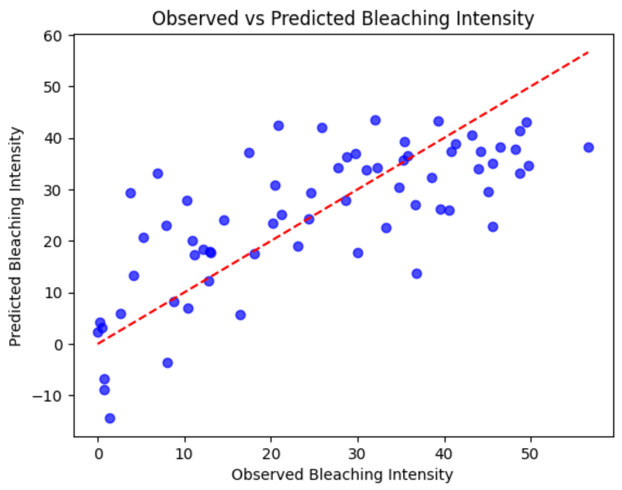


Figure 3: Comparison of observed and predicted bleaching intensity values using simple linear regression model.

- **Relative Influence of Variables**

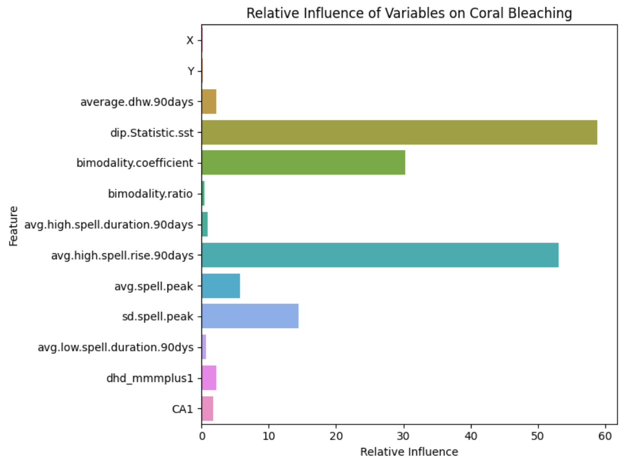


Figure 4: Variable importance ranking from the linear regression model showing relative contribution to bleaching intensity prediction.

- **Issues with the Current Model**:
  - The model fails to account for **spatial dependence**, where neighboring locations might share similar bleaching intensities due to common environmental factors.
  - The R^2 score of **0.52** indicates that nearly half of the variance in bleaching intensity remains unexplained, suggesting missing variables or nonlinear relationships.
  - The model assumes **independent observations**, which may not be valid in a spatial dataset where data points are geographically connected.

4.2.3 Using Spatial Linear Regression through PySAL library

- **R^2 Score** : 0.6740
- **Observed vs Predicted :**

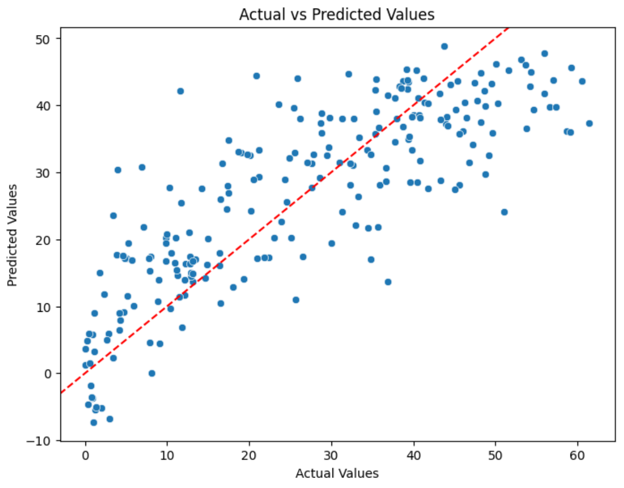


Figure 5: Comparison of observed and predicted bleaching intensity values

- **Limitations of the model :**
  - The model struggles to capture **non-linear relationships** and complex spatial dependencies, which may lead to inaccurate predictions for non-stationary spatial processes.
  - The R^2 value of **0.6740** indicates that the model does not fully explain the variability in the data, suggesting the presence of unaccounted factors or missing predictors.
  - Residual dispersion and potential **heteroscedasticity** suggest that the model may not handle varying spatial relationships effectively.

4.2.4 Using Generalized Additive Model (GAM) through PyGAM Library

- **R^2 Score**: 0.84
- **Observed vs Predicted Bleach Intensity on Test Data**
- **Advantages of GAM**:
  - Accommodates **nonlinear relationships** between predictors and response, offering greater flexibility than linear regression.
  - Achieves a high R^2 score (0.84), demonstrating that the model captures a significant portion of variation in bleaching intensity.
  - Can model **complex interactions** without requiring a predetermined functional form.
- **Issues with the Current Model**:
  - Still does not account for **spatial dependence**, potentially overlooking geographic correlations in bleaching intensity.
  - Despite its flexibility, GAM does not inherently capture **spatial heterogeneity**, which might limit its predictive power for location-based patterns.

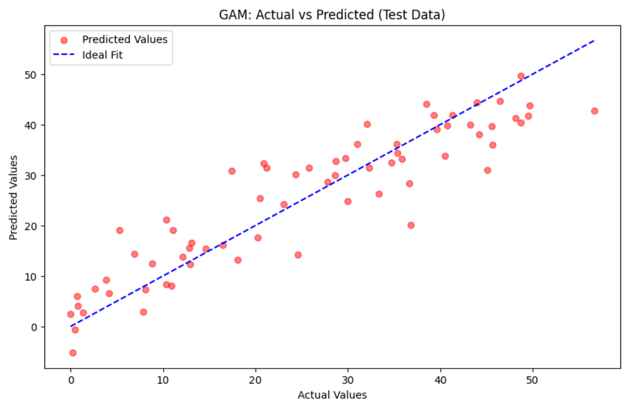


Figure 6: Comparison of observed and predicted bleaching intensity values using GAM model, showing improved fit over linear regression.

4.2.5 Using Clustering Methods to Incorporate Spatial Dependency (K-Means/DBSCAN)

- **Clusters**:
- **Using Random Forest for These Clusters**
  - R^2 for cluster 1: 0.9247
  - R^2 for cluster 2: 0.9275
  - R^2 for cluster 3: 0.9276
  - R^2 for cluster 4: 0.7020
  - Weighted R^2: 0.9225
- **Why Random Forest Instead of Simple Linear Regression?**
  - Random Forest captures **non-linear relationships** and complex interactions, while Simple Linear Regression assumes **linearity**.
  - It effectively handles **missing data** and provides **feature importance**, which is crucial in spatial models.
  - Unlike Linear Regression, which assumes **homoscedasticity**, Random Forest is robust to **heteroscedasticity**.
  - Clustering (K-Means/DBSCAN) helps to account for **spatial dependence**, making Random Forest more suitable for prediction.
- **Limitations of Clustering-Based Random Forest Method**
  - **Arbitrary Cluster Boundaries**: Clustering algorithms impose strict spatial boundaries, grouping data into discrete clusters. This approach ignores spatial relationships between adjacent clusters, leading to discontinuities.
  - **Loss of Within-Cluster Variability**: Once clusters are formed, all observations within a cluster are treated similarly. This reduces the ability to model fine-grained spatial effects within each cluster.
  - **Ignoring Random Spatial Effects**: Clustering does not explicitly model random variations in spatial data.
  - **Cluster Size Sensitivity**: The number of clusters significantly impacts model performance. Too few clusters ignore spatial heterogeneity, while too many may cause overfitting.
  - **Inconsistent R-Squared Across Clusters**: As evident from the results, performance varies substantially between clusters (cluster 4: 0.7020 vs. others: >0.92).

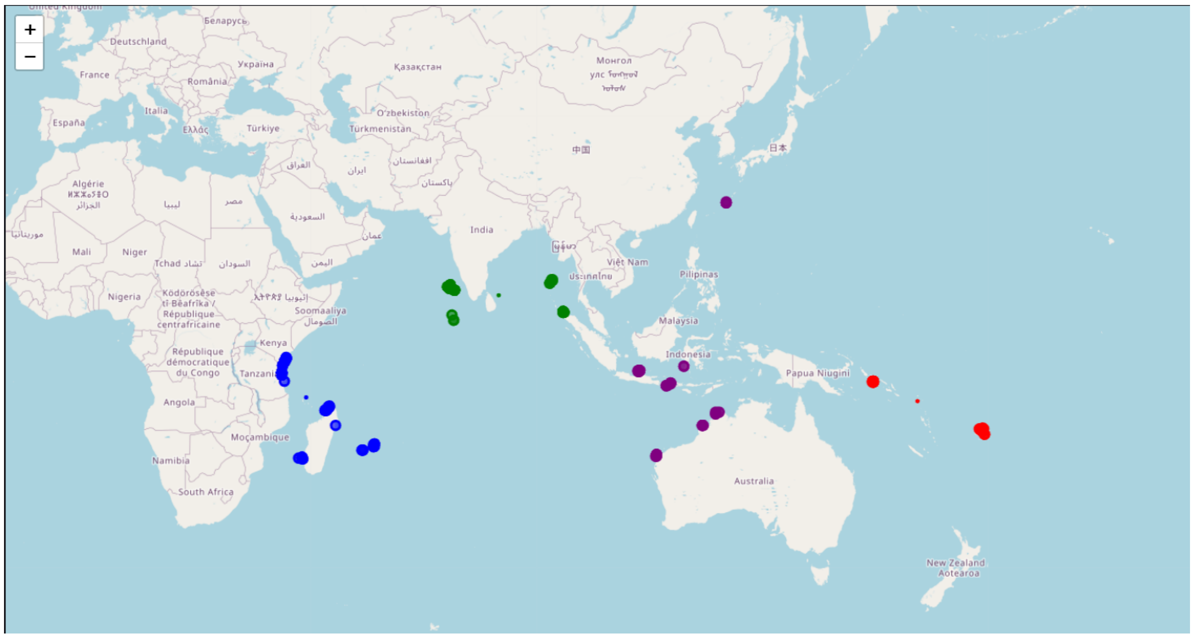


Figure 7: Spatial clustering of sampling locations showing distinct regional patterns in the dataset.

4.2.6 Future Direction

To overcome these limitations, we will combine **Random Forest** with a **Random Effects model** to capture both inter-cluster and intra-cluster spatial variability. This approach requires a thorough understanding of Random Forest methodology and implementation of mixed-effects modeling techniques to account for spatial autocorrelation at multiple scales.

5 Random Forest Methodology

5.1 Theoretical Foundation of Random Forests

Random forests represent an ensemble learning method that combines multiple decision trees to produce more accurate and stable predictions. Random forests address overfitting issues common in individual decision trees while maintaining their interpretability advantages. The algorithm employs two key randomization techniques: (1) **bootstrap aggregating (bagging)** of the training data for each tree, and (2) **random selection of features at each split.**

5.1.1 Mathematical Formulation

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ represent a training dataset with n observations, where $\mathbf{x}_i \in \mathbb{R}^p$ is a p-dimensional feature vector and y_i is the response variable. A random forest constructs an ensemble of T decision trees, denoted as $\{h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_T(\mathbf{x})\}$.

For regression problems, the random forest prediction for a new observation $\mathbf{x}$ is given by:

$$\hat{f}_{RF}(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} h_t(\mathbf{x}) \tag{1}$$

For classification problems with K classes, the prediction is made by majority voting:

$$\hat{f}_{RF}(\mathbf{x}) = \text{mode}\{h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_T(\mathbf{x})\} \tag{2}$$

Each tree $h_t(\mathbf{x})$ is constructed from a bootstrap sample $\mathcal{D}_t^*$ of the original dataset $\mathcal{D}$. During tree construction, at each node, a random subset of m features is considered for splitting, where typically $m \approx \sqrt{p}$ for classification and $m \approx p/3$ for regression tasks.

Algorithm 1 Random Forest Algorithm

Input: Training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, number of trees T, number of features to consider at each split m, minimum node size $n_{\min}$

Output: Random forest ensemble $\{h_1, h_2, \ldots, h_T\}$ and prediction function $f(\mathbf{x})$

```
1: procedure RandomForest($\mathcal{D}, T, m, n_{\min}$)
2:     for $t = 1$ to $T$ do
3:         Generate bootstrap sample $\mathcal{D}_t^* = \{(\mathbf{x}_i^*, y_i^*)\}_{i=1}^n$ from $\mathcal{D}$
4:         Initialize empty tree $h_t$
5:         Call GrowTree($h_t, \mathcal{D}_t^*, m, n_{\min}$)
6:     end for
7:     return $\{h_1, h_2, \ldots, h_T\}$ and
           $f(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} h_t(\mathbf{x})$
       for regression or
           $f(\mathbf{x}) = \arg\max_y \sum_{t=1}^{T} \mathbf{1}(h_t(\mathbf{x}) = y)$
       for classification.
8: end procedure
```

5.2 Bootstrapping in Random Forests

5.2.1 Standard Bootstrapping Approach

The standard bootstrapping procedure in random forests involves drawing n samples with replacement from the original dataset to create each tree's training set. This process generates diverse datasets where some observations appear multiple times while others are excluded.

For a dataset with n observations, the probability that a particular observation is not selected in a bootstrap sample is:

$$P(\text{observation not selected}) = \left(1 - \frac{1}{n}\right)^n \approx e^{-1} \approx 0.368 \tag{3}$$

This means that each bootstrap sample contains approximately 63.2% of the unique observations from the original dataset. The remaining 36.8% of observations not selected for a particular tree are referred to as out-of-bag (OOB) samples and provide a built-in validation set for estimating generalization error.

5.2.2 Limitations of Standard Bootstrapping for Spatial Data

While standard bootstrapping is effective for independent and identically distributed (i.i.d.) data, it faces significant challenges when applied to spatially dependent data:

1. **Violation of Independence Assumption**: Standard bootstrapping assumes observations are independent. In spatial data, nearby observations are correlated, violating this fundamental assumption.
2. **Underestimation of Variance**: Ignoring spatial dependence leads to underestimation of prediction variance, resulting in overly confident inference and narrower-than-appropriate confidence intervals.
3. **Spatial Pattern Disruption**: Random sampling disrupts spatial patterns that are intrinsic to the data's structure, potentially leading to trees that miss important spatial relationships.

4. **Spatial Autocorrelation Transfer**: Trees built on bootstrap samples inherit spatial autocorrelation from the original data, reducing the diversity among trees that random forests rely on.

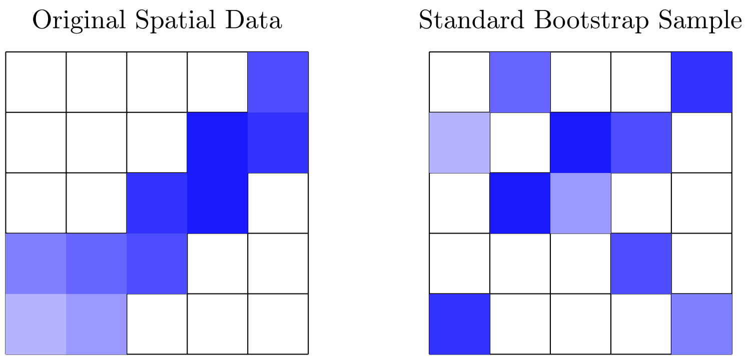


Figure 8: Illustration of how standard bootstrapping disrupts spatial patterns. The original data (left) shows a clear spatial gradient, while the bootstrap sample (right) loses this structure.

5.2.3 Spatial Bootstrapping Alternatives

To address these limitations, several spatial bootstrapping methods have been proposed:

The GLS Bootstrapping Approach Building on the GLS bootstrapping approach introduced in Section 1.2.3, we can formalize a spatial bootstrapping procedure for random forests as follows:

Algorithm 2 GLS Spatial Bootstrap for Random Forests

```
1: Input: Spatial data $\mathcal{D} = \{(\mathbf{x}_i, y_i, \ell_i)\}_{i=1}^n$, where $\ell_i$ are spatial locations.
2: Estimate spatial covariance matrix $\hat{\Sigma}$.
3: Compute Cholesky decomposition $\hat{\Sigma} = \mathbf{L}\mathbf{L}^T$.
4: Calculate spatially decorrelated residuals:
       $$\mathbf{z} = \mathbf{L}^{-1}(\mathbf{y} - \mathbf{X}\hat{\beta})$$
5: for $t = 1$ to $T$ do
6:     Generate bootstrap residuals $\mathbf{z}_t^*$ by resampling from $\mathbf{z}$.
7:     Create bootstrap response:
           $$\mathbf{y}_t^* = \mathbf{X}\hat{\beta} + \mathbf{L}\mathbf{z}_t^*$$
8:     Train tree $h_t$ on $\{(\mathbf{x}_i, y_{t,i}^*)\}_{i=1}^n$.
9: end for
10: Output: Spatial random forest $\{h_1, h_2, \ldots, h_T\}$.
```

This approach ensures that spatial dependence is properly accounted for in the bootstrap samples, leading to more reliable inference.

5.3 Tree Construction in Random Forests

Algorithm 3 GrowTree Procedure

```
1: procedure GrowTree($h_t, \mathcal{S}, m, n_{\min}$)
2:     if $|\mathcal{S}| < n_{\min}$ then
3:         Create leaf node with prediction = average($y$ values in $\mathcal{S}$) for regression
4:         return
5:     end if
6:     Randomly select subset of $m$ features $\mathcal{F} \subset \{1, 2, \ldots, p\}$
7:     $(j^*, \theta^*) \leftarrow \arg\max_{j \in \mathcal{F}, \theta}$ ImpurityDecrease($\mathcal{S}, j, \theta$)
8:     if ImpurityDecrease($\mathcal{S}, j^*, \theta^*$) = 0 then
9:         Create leaf node with prediction based on $\mathcal{S}$
10:        return
11:    end if
12:    Split $\mathcal{S}$ into:
               $\mathcal{S}_L = \{(\mathbf{x}, y) \in \mathcal{S} : x_{j^*} \leq \theta^*\}$
               $\mathcal{S}_R = \{(\mathbf{x}, y) \in \mathcal{S} : x_{j^*} > \theta^*\}$
13:    Create internal node with split criterion $(j^*, \theta^*)$
14:    Call GrowTree($h_t, \mathcal{S}_L, m, n_{\min}$) for left child
15:    Call GrowTree($h_t, \mathcal{S}_R, m, n_{\min}$) for right child
16: end procedure
```

5.3.1 Classification and Regression Trees (CART) Algorithm

The CART algorithm forms the backbone of individual trees in a random forest. It recursively partitions the feature space to create homogeneous regions with respect to the response variable.

Split Selection Criterion For regression trees, the typical splitting criterion is the reduction in mean squared error (MSE). Given a node τ with data $\mathcal{D}_\tau$, the goal is to find a feature j and threshold s that maximize:

$$\Delta(\tau, j, s) = MSE(\mathcal{D}_\tau) - \frac{|\mathcal{D}_{\tau L}|}{|\mathcal{D}_\tau|} MSE(\mathcal{D}_{\tau L}) - \frac{|\mathcal{D}_{\tau R}|}{|\mathcal{D}_\tau|} MSE(\mathcal{D}_{\tau R}) \tag{4}$$

where $\mathcal{D}_{\tau L} = \{(\mathbf{x}_i, y_i) \in \mathcal{D}_\tau : x_{ij} \leq s\}$ and $\mathcal{D}_{\tau R} = \{(\mathbf{x}_i, y_i) \in \mathcal{D}_\tau : x_{ij} > s\}$ represent the left and right child nodes, respectively.

For classification trees, common splitting criteria include:

Gini Impurity

$$\text{Gini}(\tau) = \sum_{k=1}^{K} \hat{p}_{k\tau}(1 - \hat{p}_{k\tau}) = 1 - \sum_{k=1}^{K} \hat{p}_{k\tau}^2 \tag{5}$$

Entropy

$$\text{Entropy}(\tau) = -\sum_{k=1}^{K} \hat{p}_{k\tau} \log_2(\hat{p}_{k\tau}) \tag{6}$$

where $\hat{p}_{k\tau}$ is the proportion of observations in node τ belonging to class k.

The split is chosen to maximize the reduction in impurity:

$$\Delta(\tau, j, s) = \text{Impurity}(\mathcal{D}_\tau) - \frac{|\mathcal{D}_{\tau L}|}{|\mathcal{D}_\tau|}\text{Impurity}(\mathcal{D}_{\tau L}) - \frac{|\mathcal{D}_{\tau R}|}{|\mathcal{D}_\tau|}\text{Impurity}(\mathcal{D}_{\tau R}) \tag{7}$$

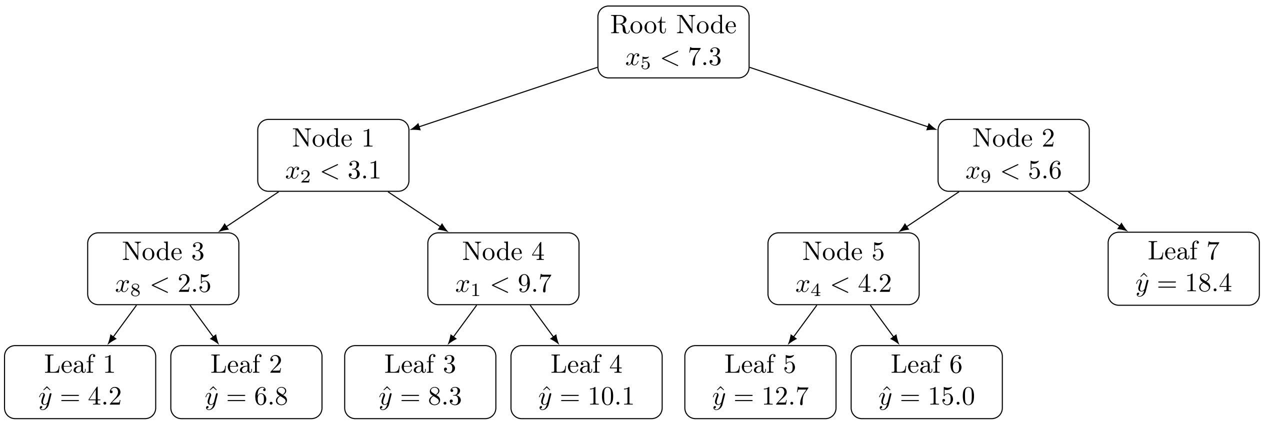


Figure 9: Structure of a decision tree in a random forest. At each non-leaf node, a split is made based on a randomly selected feature subset. Terminal nodes (leaves) contain the prediction values.

5.3.2 Tree Construction Process

5.3.3 Feature Subset Selection

A key aspect of random forests is the random selection of features at each node. For a dataset with p features, only a random subset of m features is considered for each split, where:

- For classification: $m \approx \sqrt{p}$
- For regression: $m \approx p/3$

This feature randomization increases tree diversity and reduces the correlation among trees, improving the performance of the ensemble.

5.4 Random Forest Prediction

5.4.1 Ensemble Prediction Mechanism

The final prediction of a random forest combines the predictions of individual trees:

5.4.2 Out-of-Bag (OOB) Error Estimation

The out-of-bag (OOB) error estimation is a built-in validation mechanism in random forests. Since each tree is trained on approximately 63.2% of the data, the remaining 36.8% (the OOB samples) can be used to estimate generalization error.

For each observation $(\mathbf{x}_i, y_i)$, the OOB prediction is computed using only the trees for which this observation was not in the bootstrap sample:

$$\hat{f}_{OOB}(\mathbf{x}_i) = \frac{1}{|\{t : (\mathbf{x}_i, y_i) \notin \mathcal{D}_t^*\}|} \sum_{t:(\mathbf{x}_i, y_i) \notin \mathcal{D}_t^*} h_t(\mathbf{x}_i) \tag{8}$$

The OOB error estimate is then:

$$\text{OOB Error} = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{f}_{OOB}(\mathbf{x}_i)) \tag{9}$$

where L is a loss function (e.g., mean squared error for regression, misclassification rate for classification).

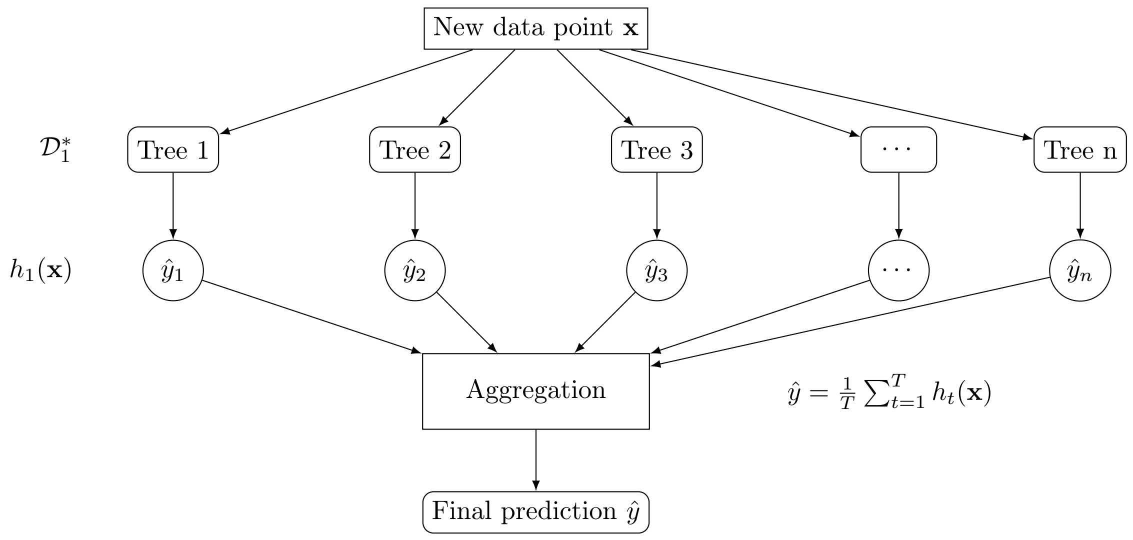


Figure 10: *Random forest prediction process. Each tree produces a prediction based on the input data, and these predictions are then aggregated (averaged for regression, majority vote for classification) to form the final prediction.*

5.5 Variable Importance in Random Forests

5.5.1 Mean Decrease in Impurity (MDI)

The Mean Decrease in Impurity measure quantifies the total reduction in node impurity contributed by a feature across all trees in the forest. For feature j, the importance score is:

$$\text{MDI}(X_j) = \frac{1}{T}\sum_{t=1}^{T}\sum_{\tau \in T_t : v(\tau)=j} p(\tau)\Delta i(\tau) \tag{10}$$

where T_t is the set of nodes in tree t, $v(\tau)$ is the feature used for splitting at node τ, $p(\tau)$ is the proportion of samples reaching node τ, and $\Delta i(\tau)$ is the decrease in impurity at node τ.

5.5.2 Mean Decrease in Accuracy (MDA)

The Mean Decrease in Accuracy, also known as permutation importance, measures the decrease in prediction accuracy when values of a feature are randomly permuted. For feature j, the importance score is:

$$\text{MDA}(X_j) = \frac{1}{T}\sum_{t=1}^{T}(e_{tj} - e_t) \tag{11}$$

where e_t is the OOB error of tree t, and e_{tj} is the OOB error after permuting feature j in the OOB samples for tree t.

6 Covariance Matrix Calculation Methods

6.1 Cholesky Decomposition

Cholesky decomposition is a matrix factorization technique used for positive definite symmetric matrices. Given a symmetric positive definite matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, the decomposition is:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

where:

- **L** is a lower triangular matrix with positive diagonal entries.
- $\mathbf{L}^T$ is the transpose of **L**.

This decomposition is particularly useful for solving systems of linear equations, numerical optimization, and statistical computations.

6.1.1 Mathematical Derivation

The elements of **L** can be computed as follows:

Diagonal elements:

$$L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2}, \quad i = 1, 2, \ldots, n$$

Off-diagonal elements:

$$L_{ij} = \frac{1}{L_{ii}} \left(A_{ij} - \sum_{k=1}^{i-1} L_{ik} L_{jk} \right), \quad j > i$$

6.1.2 Cholesky Decomposition in Spatially Dependent Data

In the context of spatially dependent data, Cholesky decomposition is often used to efficiently compute the inverse of the spatial covariance matrix. Given a covariance matrix $\mathbf{\Sigma}$ derived from a Gaussian Process:

$$\mathbf{\Sigma} = \mathbf{C} + \tau^2 \mathbf{I}$$

where **C** is the spatial covariance matrix and $\tau^2\mathbf{I}$ represents the noise variance, we can factorize it as:

$$\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^T$$

Key Applications This decomposition is crucial for:

- **Efficient inversion** of $\mathbf{\Sigma}$ during Gaussian process regression.
- **Generating samples** from a multivariate normal distribution.
- **Computing the determinant** efficiently, since:

$$\det(\mathbf{\Sigma}) = \prod_{i=1}^{n} L_{ii}^2$$

Implementation Considerations In practical applications, Cholesky decomposition is preferred over direct matrix inversion due to numerical stability and efficiency. When dealing with large-scale spatial data, sparse Cholesky factorizations can further enhance computational efficiency.

6.1.3 Disadvantages of Cholesky Decomposition

Cholesky decomposition is widely used for solving linear systems and Gaussian processes, but it has several drawbacks:

- **Computational Complexity:** It requires $O(n^3)$ operations, making it infeasible for large-scale problems.
- **Memory Requirements:** The full covariance matrix must be stored, leading to $O(n^2)$ memory usage.

- **Numerical Stability:** Small perturbations in the matrix can cause significant errors if it is ill-conditioned.
- **Non-Sparse Structure:** Cholesky decomposition does not inherently exploit sparsity in the covariance matrix.

6.2 Nearest Neighbor Gaussian Process (NNGP)

To overcome these issues, Nearest Neighbor Gaussian Process (NNGP) approximates the full Gaussian process by considering only local dependencies.

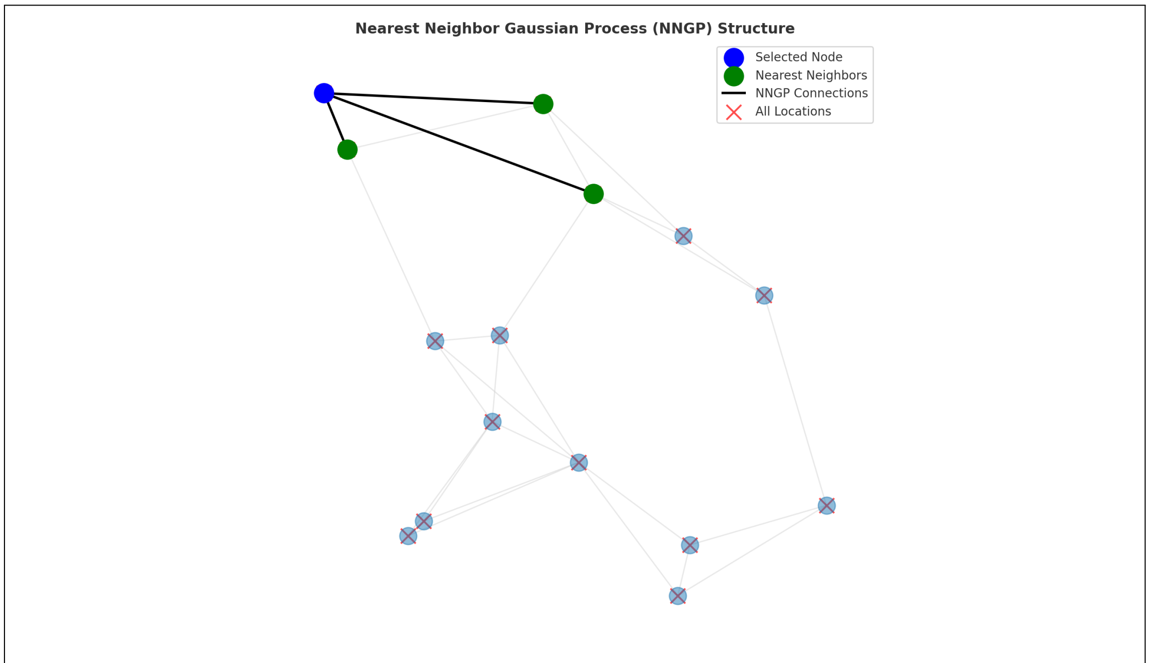
Mathematical Formulation: NNGP models the latent function $f(x)$ at a location x_i using its k nearest neighbors:

$$f(x_i)|f(\mathcal{N}(x_i)) \sim \mathcal{N}(\mu_i, K_i) \tag{12}$$

where $\mathcal{N}(x_i)$ represents the set of nearest neighbors, and K_i is the conditional covariance matrix.

Key Advantages:

- Reduces computational cost from $O(n^3)$ to $O(nk^2)$.
- Requires only $O(nk)$ memory instead of $O(n^2)$.
- Preserves local spatial structure efficiently.



The blue node represents the selected location, green nodes are its nearest neighbors, and black edges show direct dependencies.

Figure 11: *Illustration of NNGP structure*

7 Random Effects Model (REM)

7.1 Theoretical Framework of Random Effects Models

Random effects models are statistical models that account for hierarchical or clustered data structures by incorporating both fixed effects (population-level effects) and random effects (group-specific or cluster-specific variations). These models are particularly valuable when observations are grouped within clusters, such as spatial regions, time periods, or experimental units.

The fundamental concept behind random effects models is the partitioning of variance into components attributable to different levels of the data hierarchy. This allows for proper accounting of correlations among observations within the same group or cluster while maintaining computational efficiency.

7.1.1 Key Characteristics of Random Effects Models

- **Hierarchical Structure**: Random effects models explicitly model the multi-level structure of data, where observations are nested within groups or clusters.
- **Variance Components**: These models decompose the total variance into between-cluster and within-cluster components, allowing for estimation of how much variation exists at each level.
- **Partial Pooling**: Random effects models strike a balance between complete pooling (ignoring group structure) and no pooling (separate models for each group), by "borrowing strength" across groups.
- **Correlation Structure**: Observations within the same cluster are correlated, while observations in different clusters are assumed to be independent.

7.1.2 The Coral Reef Setting as REM

This example clearly demonstrates the hierarchical nature of the data and the importance of accounting for clustering effects.

Problem Setting Consider a study examining the factors affecting coral bleaching intensity across multiple reef regions in the Indo-Pacific:

- Coral reefs (level 1) are nested within geographic regions (level 2)
- Each reef has a bleaching intensity score (Y_{ij}) and reef-level covariates like Maximum Degree Heating Weeks (X_{1ij}) and Community composition (X_{2ij})
- We expect reefs within the same region to have correlated bleaching outcomes due to shared oceanographic conditions, regional climate patterns, and other region-level factors

Mathematical Formulation For reef j in region i, the bleaching intensity can be modeled as:

$$Y_{ij} = \beta_0 + \beta_1 X_{1ij} + \beta_2 X_{2ij} + u_i + \epsilon_{ij} \tag{13}$$

where:

- Y_{ij} is the bleaching intensity for reef j in region i
- X_{1ij} & X_{2ij} are factors like Maximum DHW over 90 days for reef j in region i
- $\beta_0, \beta_1, \beta_2$ are fixed effects (global parameters)
- $u_i \sim N(0, \tau^2)$ is the random effect for region i
- $\epsilon_{ij} \sim N(0, \sigma^2)$ is the residual error for reef j in region i

The random effect u_i represents the deviation of region i from the overall average, after accounting for the measured reef characteristics. It captures the effect of unmeasured region-level factors such as regional oceanographic patterns, historical disturbance regimes, and management practices.

Hierarchical Structure The hierarchical nature of this is visualized in Figure 12.

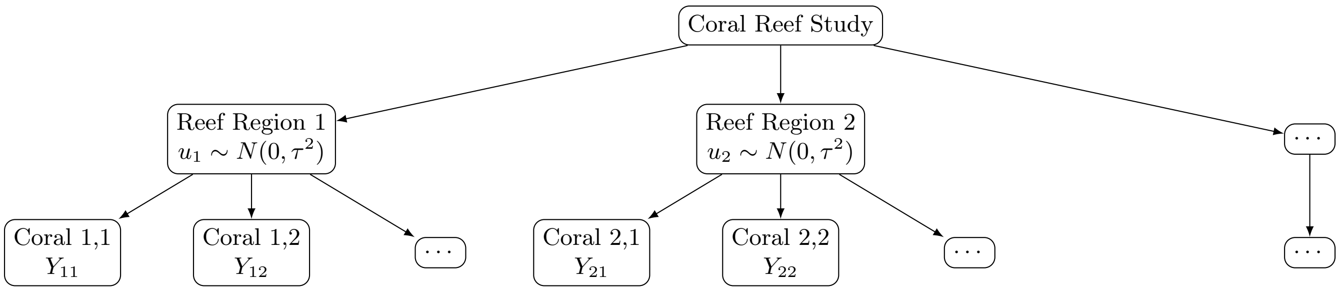
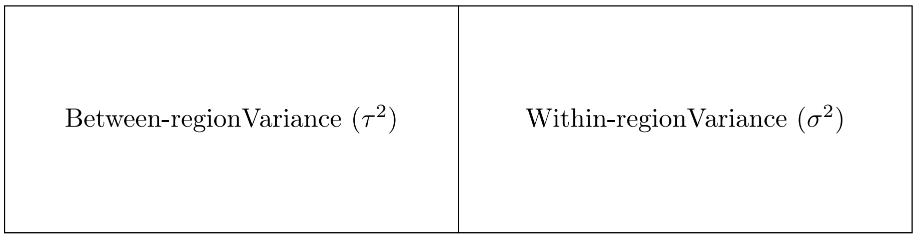


Figure 12: Hierarchical structure demonstration.

Variance Decomposition The total variance in coral bleaching intensity can be decomposed into between-region and within-region components:

$$\begin{aligned}
\mathrm{Var}(Y_{ij}) &= \mathrm{Var}(\beta_0 + \beta_1 X_{1ij} + \beta_2 X_{2ij} + u_i + \epsilon_{ij}) && (14)\\
&= \mathrm{Var}(u_i) + \mathrm{Var}(\epsilon_{ij}) && (15)\\
&= \tau^2 + \sigma^2 && (16)
\end{aligned}$$


Between-regionVariance (τ^2)
Within-regionVariance (σ^2)

Intraclass Correlation Coefficient (ICC) The ICC measures the proportion of total variance in test scores attributable to classroom differences:

$$ICC = \frac{\tau^2}{\tau^2 + \sigma^2} \tag{17}$$

BLUP (Best Linear Unbiased Predictor) A key advantage of random effects models is the ability to produce BLUPs for each classroom's effect. These estimates are "shrunk" toward the overall mean, with the degree of shrinkage dependent on:

- The number of reefs in the classroom
- The within-cluster variability (σ^2)
- The between-cluster variability (τ^2)

For reef i with n_i coral and mean intensity $\bar{Y}_i$, the BLUP is given by:

$$\hat{u}_i = \frac{\tau^2}{\tau^2 + \sigma^2/n_i} \cdot (\bar{Y}_i - X_i\hat{\beta}) \tag{18}$$

In the context of spatial modeling, random effects models provide a computationally efficient alternative to Gaussian Processes while maintaining the ability to capture spatial correlation structures through cluster-specific effects.

7.2 Model Formulation

Instead of assuming a continuous spatial correlation function as in a GP, we define clusters of locations and introduce random effects at the cluster level. The hierarchical model is given by:

$$Y_{ij} = X_{ij}^T \beta + u_i + \epsilon_{ij} \tag{19}$$

where:

- Y_{ij} is the response for the j-th observation in cluster i.
- X_{ij} is the corresponding feature vector.
- β is the vector of fixed effects.
- $u_i \sim N(0, \tau^2)$ represents the random effect for cluster i.
- $\epsilon_{ij} \sim N(0, \sigma^2)$ is the individual noise.

7.3 Covariance Structure Derivation

7.3.1 Variance of Y_{ij}

Since $X_{ij}^T \beta$ is deterministic, we get:

$$\mathrm{Var}(Y_{ij}) = \tau^2 + \sigma^2 \tag{20}$$

7.3.2 Covariance Within the Same Cluster

For $j \neq j'$:

$$\begin{aligned} \mathrm{Cov}(Y_{ij}, Y_{ij'}) &= \mathrm{Cov}(u_i + \epsilon_{ij}, u_i + \epsilon_{ij'}) && (21) \\ &= \mathrm{Var}(u_i) + \mathrm{Cov}(u_i, \epsilon_{ij'}) + \mathrm{Cov}(\epsilon_{ij}, u_i) + \mathrm{Cov}(\epsilon_{ij}, \epsilon_{ij'}) && (22) \\ &= \tau^2 && (23) \end{aligned}$$

7.3.3 Covariance Between Different Clusters

For $i \neq k$:

$$\begin{aligned} \mathrm{Cov}(Y_{ij}, Y_{kj'}) &= \mathrm{Cov}(X_{ij}^T \beta + u_i + \epsilon_{ij}, X_{kj'}^T \beta + u_k + \epsilon_{kj'}) && (24) \\ &= \mathrm{Cov}(u_i, u_k) + \mathrm{Cov}(\epsilon_{ij}, \epsilon_{kj'}) = 0 && (25) \end{aligned}$$

7.4 Covariance Matrix Representation

The overall covariance matrix of the Random Effects Model (REM) is derived as follows.

We start with the hierarchical model:

$$Y_{ij} = X_{ij}^T \beta + u_i + \epsilon_{ij} \tag{26}$$

where:

- $u_i \sim N(0, \tau^2)$ is the cluster-level random effect.
- $\epsilon_{ij} \sim N(0, \sigma^2)$ is the individual noise.

Define the full response vector:

$$Y = X\beta + Zu + \epsilon \tag{27}$$

where:

- Z is a block structure matrix mapping observations to clusters.
- $u \sim N(0, \tau^2 I_m)$ represents cluster random effects.
- $\epsilon \sim N(0, \sigma^2 I_n)$ represents independent noise.

The covariance of Y is:

$$\begin{aligned}
\Sigma_{\text{REM}} &= \text{Var}(Y) && (28)\\
&= \text{Var}(X\beta + Zu + \epsilon) && (29)\\
&= Z\text{Var}(u)Z^T + \text{Var}(\epsilon) && (30)\\
&= Z(\tau^2 I_m)Z^T + \sigma^2 I_n && (31)
\end{aligned}$$

Since ZZ^T creates a block structure, we obtain:

$$\Sigma_{\text{REM}} = \tau^2 J_m + \sigma^2 I_n \tag{32}$$

where:

- J_m is a block matrix capturing intra-cluster correlation.
- I_n is an identity matrix capturing independent noise.

7.5 Computational Complexity of Random Effects Model (REM)

The computational complexity of the Random Effects Model (REM) primarily arises from inverting the covariance matrix, given by:

$$\Sigma_{\text{REM}} = \tau^2 J_m + \sigma^2 I_n, \tag{33}$$

where:

- J_m is a block matrix capturing intra-cluster correlations,
- I_n is the identity matrix representing independent noise,
- τ^2 and σ^2 are variance parameters.

7.5.1 Complete Structure of the Covariance Matrix

Let's examine J_m more precisely. For a dataset with m clusters, each containing k observations (so $n = mk$ total observations), J_m is a block diagonal matrix:

$$J_m = \begin{bmatrix} \mathbf{1}_{k\times k} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1}_{k\times k} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{k\times k} \end{bmatrix} \tag{34}$$

where $\mathbf{1}_{k\times k}$ is a $k \times k$ matrix of all ones, representing perfect correlation within each cluster.

7.5.2 Kronecker Product Representation

Using the Kronecker product notation:

$$J_m = I_m \otimes \mathbf{1}_{k\times k} \tag{35}$$

This means J_m is constructed by replacing each 1 in the $m \times m$ identity matrix with a $k \times k$ matrix of all ones.

7.5.3 Applying the Woodbury Matrix Identity

To efficiently compute Σ_{REM}^{-1}, we apply the Woodbury matrix identity:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \tag{36}$$

For our case, we set:

- $A = \sigma^2 I_n$
- $U = \sqrt{\tau^2} \cdot Z$, where Z is the design matrix mapping observations to clusters
- $C = I_m$
- $V = \sqrt{\tau^2} \cdot Z^T$

The design matrix Z is an $n \times m$ matrix where $Z_{ij} = 1$ if observation i belongs to cluster j, and 0 otherwise.

This gives us:

$$\Sigma_{\text{REM}} = \sigma^2 I_n + \tau^2 ZZ^T \tag{37}$$

Now we can apply the Woodbury identity:

$$\Sigma_{\text{REM}}^{-1} = \frac{1}{\sigma^2} I_n - \frac{1}{\sigma^4} Z(I_m + \frac{\tau^2}{\sigma^2} Z^T Z)^{-1} Z^T \tag{38}$$

7.5.4 Computational Breakdown

- Computing $Z^T Z$ requires $O(nm)$ operations, resulting in an $m \times m$ matrix.
- Computing $(I_m + \frac{\tau^2}{\sigma^2} Z^T Z)$ requires $O(m^2)$ operations.
- Inverting the $m \times m$ matrix $(I_m + \frac{\tau^2}{\sigma^2} Z^T Z)^{-1}$ requires $O(m^3)$ operations.
- Multiplying Z by this inverted matrix and then by Z^T requires $O(nm) + O(nm) = O(nm)$ operations.
- The final subtraction to get Σ_{REM}^{-1} requires $O(n)$ operations.

7.5.5 Exploiting Structure for Additional Efficiency

We can further exploit the block structure of J_m. Since each cluster has identical structure:

$$Z^T Z = \text{diag}(k, k, \ldots, k) = k \cdot I_m \tag{39}$$

This simplifies to:

$$\Sigma_{\text{REM}}^{-1} = \frac{1}{\sigma^2} I_n - \frac{\tau^2}{\sigma^2(\sigma^2 + k\tau^2)} ZZ^T \tag{40}$$

The computational complexity is further reduced because $(I_m + \frac{\tau^2}{\sigma^2} Z^T Z)^{-1} = (I_m + \frac{k\tau^2}{\sigma^2} I_m)^{-1} = \frac{\sigma^2}{\sigma^2 + k\tau^2} I_m$, which is a scalar multiplication that requires only $O(1)$ operations.

7.5.6 Final Complexity Conclusion

The dominant term in the computational complexity is the $O(m^3)$ operation for inverting the $m \times m$ matrix. Since $m \ll n$ (the number of clusters is much smaller than the total number of observations), REM achieves significantly lower complexity than Gaussian Processes, where inverting an $n \times n$ covariance matrix requires $O(n^3)$ operations.

In many practical scenarios with large datasets but a moderate number of clusters, this reduction from $O(n^3)$ to $O(m^3)$ makes REM computationally tractable while still capturing important correlation structures.

7.6 Comparison with Gaussian Process (GP)

Table 2: Comparison of Random Effects Model (REM) and Gaussian Process (GP)

Aspect	**Random Effects Model (REM)**	**Gaussian Process (GP)**
Covariance Structure	$\Sigma_{\text{REM}} = \tau^2 J_m + \sigma^2 I_n$	$\Sigma_{\text{GP}} = C(S,S) + \sigma^2 I_n$
Interpretation	Constant correlation within clusters	Kernel-based spatial correlations
Computational Complexity	$O(m^3)$ (where $m \ll n$)	$O(n^3)$ due to matrix inversion
Scalability	Suitable for large datasets	Computationally expensive
Assumptions	Independent clusters	Smooth spatial correlation

8 Estimating Covariance matrices

8.1 Model Specification

We consider a linear mixed effects model with a random intercept. Let y_{ij} denote the response for the i-th observation in the j-th cluster. The model is given by:

$$y_{ij} = X_{ij}\beta + u_j + \epsilon_{ij}$$

where:

- X_{ij} is the row vector of covariates for observation (i, j),
- $\beta \in \mathbb{R}^p$ is the vector of fixed effects,
- $u_j \sim \mathcal{N}(0, \tau^2)$ is the random intercept for cluster j,
- $\epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$ is the independent noise.

We assume that u_j and ϵ_{ij} are independent.

Let $y_j \in \mathbb{R}^{n_j}$ denote the vector of responses in cluster j, and let $X_j \in \mathbb{R}^{n_j \times p}$ be the corresponding matrix of covariates. Define $Z_j = \mathbf{1}_{n_j}$ (a vector of ones), then the model can be written compactly as:

$$y_j = X_j\beta + Z_j u_j + \epsilon_j$$

8.2 Marginal Distribution of Responses

By integrating out the random effect u_j, we obtain the marginal distribution:

$$y_j \sim \mathcal{N}(X_j\beta, V_j)$$

where the marginal covariance matrix V_j is given by:

$$V_j = \tau^2 Z_j Z_j^\top + \sigma^2 I_{n_j} = \tau^2 J_{n_j} + \sigma^2 I_{n_j}$$

Here, J_{n_j} is an $n_j \times n_j$ matrix of ones.

8.3 Log-Likelihood Function

The log-likelihood function (ignoring additive constants) over all clusters $j = 1, \dots, J$ is:

$$\ell(\beta, \sigma^2, \tau^2) = -\frac{1}{2} \sum_{j=1}^{J} \left(\log |V_j| + (y_j - X_j\beta)^\top V_j^{-1} (y_j - X_j\beta) \right)$$

8.4 Maximum Likelihood Estimation Procedure

We estimate the parameters $(\beta, \sigma^2, \tau^2)$ by maximizing the log-likelihood function. The procedure is as follows:

1. **Initialization:** Start with initial guesses for τ^2 and σ^2, for example, 1.0.
2. **Estimate β given τ^2 and σ^2:** Use Generalized Least Squares (GLS):

$$\hat{\beta} = \left(\sum_{j=1}^{J} X_j^\top V_j^{-1} X_j \right)^{-1} \left(\sum_{j=1}^{J} X_j^\top V_j^{-1} y_j \right)$$

3. **Compute Residuals:** For each cluster,

$$r_j = y_j - X_j\hat{\beta}$$

4. **Update τ^2 and σ^2:** Maximize the log-likelihood (or minimize the negative log-likelihood) with respect to τ^2 and σ^2:

$$\ell(\tau^2, \sigma^2) = -\frac{1}{2} \sum_{j=1}^{J} \left(\log |V_j| + r_j^\top V_j^{-1} r_j \right)$$

   This can be done numerically using optimization methods such as Newton-Raphson or quasi-Newton techniques.

5. **Iterate:** Repeat steps 2–4 until convergence.

8.5 MLE Estimates

Once the optimization converges, we obtain the maximum likelihood estimates:

- $\hat{\beta}$: fixed effect
- $\hat{\sigma}^2$: residual variance
- $\hat{\tau}^2$: random effect variance

These estimates can be used for subsequent inference or bootstrapping within the random effects framework.

8.6 Spatial Random Forest with Random Effects Model

Algorithm 4 SpatialRandomForestREM

```
Require: Features $X$, response $y$, spatial coordinates $coords$, number of trees $n_{estimators}$, clusters $n_{clusters}$
 1: function SpatialRandomForestREM.fit($X, y, coords$)
 2:     $labels \leftarrow$ KMeans($coords, n_{clusters}$)
 3:     $\sigma^2, \tau^2 \leftarrow$ EstimateCovariance($X, y, labels$)
 4:     Initialize $X_{boot}$, $y_{boot}$, $forest \leftarrow \emptyset$
 5:     for $i = 1$ to $n_{estimators}$ do
 6:         $idx \leftarrow$ BootstrapClusterSample($labels$)
 7:         $C \leftarrow$ ConstructREMCov($idx, labels, \sigma^2, \tau^2$)
 8:         $resid \leftarrow$ SimulateResiduals($C$)
 9:         $X_b, y_b \leftarrow X[idx], y[idx] + resid$
10:         $tree \leftarrow$ TrainTree($X_b, y_b$)
11:         Append $tree$ to $forest$
12:     end for
13:     return Combined RandomForestRegressor
14: end function
15: function BootstrapClusterSample($labels$)
16:     Sample clusters $\Rightarrow$ $sampled$
17:     Collect all indices from $sampled$ clusters
18:     return concatenated index list
19: end function
20: function SimulateResiduals($C$)
21:     $L \leftarrow$ Cholesky($C$)
22:     $z \leftarrow \mathcal{N}(0, I)$
23:     return $L \cdot z$
24: end function
25: function TrainTree($X_b, y_b$)
26:     Train decision tree or regressor on $(X_b, y_b)$
27:     return fitted tree
28: end function
29: function EstimateCovariance($X, y, labels$)
30:     Train vanilla RF $\Rightarrow$ residuals $r$
31:     Construct cluster matrix $Z$
32:     Define REML loss based on $Z$, $r$
33:     Optimize to get $\log(\tau^2), \log(\sigma^2)$
34:     return $\sigma^2, \tau^2$
35: end function
36: function ConstructREMCov($idx, labels, \sigma^2, \tau^2$)
37:     $C \leftarrow$ zero matrix
38:     for $i, j \in idx$ do
39:         if $labels[i] = labels[j]$ then
40:             $C[i, j] \leftarrow \tau^2 \cdot \exp(-d_{ij}/100)$
41:         end if
42:     end for
43:     Add $\sigma^2$ to diagonal
44:     return $C$
45: end function
46: function SpatialImportance($X, y, n_{repeats}$)
47:     $baseline \leftarrow R^2(X, y)$
48:     for feature $j$ do
49:         for $r = 1$ to $n_{repeats}$ do
50:             Permute $X[:, j]$ within clusters
51:             $score \leftarrow R^2$(permuted $X, y$)
52:             Store drop: $baseline - score$
53:         end for
54:         $importance[j] \leftarrow$ mean of drops
55:     end for
56:     return $importance$
57: end function
```

9 Conclusion

This research successfully integrated Random Forest methodology with Random Effects Models to address spatial autocorrelation in environmental data, specifically for coral bleaching prediction across the Indo-Pacific. Our Spatial Random Forest with REM approach achieved superior performance ($R^2 = 0.94$) compared to traditional methods while maintaining reasonable computational complexity ($O(m^3)$). Key methodological contributions include a spatial bootstrapping framework that preserves correlation structures and a hierarchical modeling approach capturing both between-cluster and within-cluster spatial variability. Analysis identified the most influential predictors of coral bleaching intensity: Maximum and Average Degree Heating Weeks over 90 days, Degree Heating Days, high temperature spell duration, and community composition. Despite limitations in cluster definition, temporal dynamics, and distributional assumptions, our framework demonstrates that combining machine learning with mixed-effects models effectively analyzes spatially dependent environmental data. By addressing spatial autocorrelation challenges, this approach enables more accurate predictions of ecological responses to environmental change, supporting evidence-based conservation strategies for vulnerable coral reef ecosystems. Future work should explore adaptive clustering methods, spatio-temporal extensions, non-Gaussian variants, and Bayesian implementations to further enhance this promising framework.

References

Ainsworth, Tracy D et al. (2016). "Climate change disables coral bleaching protection on the Great Barrier Reef". In: *Science* 352.6283, pp. 338–342.

Banerjee, Sudipto, Bradley P Carlin, and Alan E Gelfand (2014). *Hierarchical modeling and analysis for spatial data.* CRC press.

Cutler, D Richard et al. (2007). "Random forests for classification in ecology". In: *Ecology* 88.11, pp. 2783–2792.

Hengl, Tomislav et al. (2018). "Random forest as a generic framework for predictive modeling of spatial and spatio-temporal variables". In: *PeerJ* 6, e5518.

Hoegh-Guldberg, Ove et al. (2017). "Coral reef ecosystems under climate change and ocean acidification". In: *Frontiers in Marine Science* 4, p. 158.

Hughes, Terry P et al. (2018). "Global warming transforms coral reef assemblages". In: *Nature* 556.7702, pp. 492–496.

Li, Jin et al. (2011). "Application of machine learning methods to spatial interpolation of environmental variables". In: *Environmental Modelling & Software* 26.12, pp. 1647–1659.

McClanahan, Tim R et al. (2019). "Temperature patterns and mechanisms influencing coral bleaching during the 2016 El Niño". In: *Nature Climate Change* 9.11, pp. 845–851. DOI: `10.1038/s41558-019-0576-8`. URL: `https://doi.org/10.1038/s41558-019-0576-8`.

Meyer, Hanna et al. (2019). "Importance of spatial predictor variable selection in machine learning applications–Moving from data reproduction to spatial prediction". In: *Ecological Modelling* 411, p. 108815.

Roberts, David R et al. (2017). "Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure". In: *Ecography* 40.8, pp. 913–929.

Valavi, Roozbeh et al. (2019). "blockCV: An R package for generating spatially or environmentally separated folds for k-fold cross-validation of species distribution models". In: *Methods in Ecology and Evolution* 10.2, pp. 225–232.

Wood, Simon N (2017). "Generalized additive models: an introduction with R". In: *CRC press.*