Kriging In Stan: Comparison of Spatial Interpolation Models on Crime in Vancouver

Zelalem Araya (92797935) & Youjung Kim (3876269)

2024-04-21

Repository Link:

https://github.com/eunicekim919/STAT-447c-project.git

Introduction

In recent years, densely populated regions worldwide have experienced fluctuating crime rates due to various economic and environmental factors such as income, job availability, and climate. Historically, Vancouver has consistently had higher than expected about its population size and the average Canadian crime rate, including higher rates of violent crime. (Andresen, 2017; Statistics Canada, 2023) These rising crime rates prompt concerns about public safety, and how to appropriately allocate resources to adequately deal with issues unique to each area. Effective spatial analysis of crime data is crucial for understanding these patterns, predicting future occurrences, and implementing the right preventative measures. When reports are only available for certain regions, being able to interpolate available data spatially could provide important information without needing to expend unnecessary resources. Among the various statistical techniques, kriging offers a robust method of spatial interpolation using Bayesian optimization over a Gaussian process.

This study aims to analyze and compare different kriging models for interpolating crime density in Vancouver using reported crime data from the Vancouver Police Department GeoDASH Open Data, a source that provides aggregate crime data from the crime reports made in the City of Vancouver in 2022. (Vancouver Police Department, 2024). Both Stan and R are used to implement and assess the model, with the performance evaluated through Leave-One-Out Cross-Validation (LOO-CV). This technique assesses the model's predictive performance by leaving one observation out at a time and predicting its value using the rest of the data. Determining the most appropriate kriging model for this type of data & analysis has the potential to contribute to the greater field of geospatial crime analysis as a whole, informing policy decisions and strategic planning for crime reporting and urban safety management. Exploring a variety of kriging approaches, this paper also provides a better understanding of the advantages and disadvantages of the various kriging models. Kriging & Bayesian Foundations

Background

Kriging is the overarching term for a group of geostatistical techniques used for spatial interpolation, which estimate the values at unknown points based on the values of known points based on the values of the known points nearby - based on applications of Tobler's first law. The foundation of kriging lies in its ability to model spatial autocorrelation through a semivariogram, a function that quantifies the strength of statistical correlation as a function of distance. This works by assuming that the distance between sample points reflects a quantifiable spatial correlation, that can then be used to interpolate values at unsampled locations. Kriging functions as a kind of Bayesian linear estimator, it works by incorporating some prior knowledge about the spatial processes, and then updating with the observed data to make predictions - the unknown values considered as random variables.

Mathematical Explanation of Kriging

Kriging leverages the spatial covariance among data points to make predictions at unsampled locations. The key to kriging is the assumption that the spatial process underlying the observed data is a realization of a Gaussian process, consistent among all types of kriging methods. This process is governed by a mean function (often taken as constant or linear) and a covariance function, which quantifies how observations at different locations relate to one another based on their spatial separation. Below is a general mathematical overview of our current understanding of the method's components. (Handcock & Stein, 1993; Webster & Oliver, 2007; Wiskotten et al., 2023)

The observed data $\mathbf{Z}$ are modeled as a realization of a Gaussian process:

$$Z(\mathbf{s}) = \mu(\mathbf{s}) + \epsilon(\mathbf{s})$$

where:

- $\mathbf{s}$ denotes a spatial location.
- $\mu(\mathbf{s})$ is the mean function, often assumed to be constant α across the space.
- $\epsilon(\mathbf{s})$ represents the spatially correlated random error, assumed to follow a multivariate normal distribution.

The core of a kriging model lies in its specification of the spatial covariance. A common choice is the exponential covariance function:

$$cov(Z(\mathbf{s}_i), Z(\mathbf{s}_j)) = \sigma^2 \exp\left(-\frac{\|\mathbf{s}_i - \mathbf{s}_j\|}{\rho}\right)$$

where:

- $\mathbf{s}_i - \mathbf{s}_j$ is the Euclidean distance between locations $\mathbf{s}_i$ and $\mathbf{s}_j$.
- σ^2 is the process variance.
- ρ is the range parameter, influencing how quickly correlations decay with distance.

The prediction of $Z(\mathbf{s}_0)$ at a new location $\mathbf{s}_0$ using observed data $\mathbf{Z}$ at known locations $\mathbf{s}_1, \ldots, \mathbf{s}_N$ is given by:

$$\hat{Z}(\mathbf{s}_0) = \mu + \mathbf{k}^T \mathbf{K}^{-1}(\mathbf{Z} - \mu\mathbf{1})$$

where:

- μ is the assumed mean of the process.
- $\mathbf{Z}$ is the vector of observed values at the known locations.
- $\mathbf{k}$ is the vector of covariances between the new location $\mathbf{s}_0$ and each of the known locations.
- $\mathbf{K}$ is the covariance matrix among the observed data points.
- $\mathbf{1}$ is a vector of ones (assuming a constant mean).

The components of $\mathbf{k}$ and $\mathbf{K}$ are computed using some covariance function:

$$\mathbf{k}[i] = \sigma^2 \exp\left(-\frac{\|\mathbf{s}_0 - \mathbf{s}_i\|}{\rho}\right)$$

$$\mathbf{K}[i,j] = \sigma^2 \exp\left(-\frac{\|\mathbf{s}_i - \mathbf{s}_j\|}{\rho}\right) + \sigma^2_{\text{err}}\delta_{ij}$$

with δ_{ij} being the Kronecker delta, which is 1 if $i = j$ and 0 otherwise, equivalent to the identity matrix in practice.

Finally, the model can also incorporate measurement error σ^2_{err}, which adds to the diagonal elements of the covariance matrix $\mathbf{K}$, representing error variance that is independent at each observed location.

Other Spatial Interpolation Methods

Kriging is one of the most sophisticated geostatistical models, but others serve similar tasks - such as inverse distance weighting which computes the weighted average of nearby observed values or splines, which fit some smooth mathematical function that passes through the data points. However, kriging is often better for more complex given its semivariogram and allowing measures of uncertainty for those estimates.

Types of Kriging

Since Kriging is highly adaptive, various kinds can be used in different situations. Simple Kriging assumes a known constant mean throughout the entire region. Ordinary Kriging is the most common, it does not assume a known mean and instead estimates it based on available data. Universal kriging pushes the mean estimation further by modelling the mean as a polynomial function of coordinates, best for analyzing trends across a region. (Mooney et al., 2018). Bayesian Kriging models incorporate spatial modeling utilizing a parametric function of distance that includes the longitude and latitude as random variables. (Zhang & Du, 2019) Kriging Model Comparisons

While there are many types of kriging, determining which type of kriging is most appropriate for given situations is a complex task. Wiskotten et al. discuss the use of ordinary kriging and Bayesian kriging in the context of radiological characterization for nuclear facility decommissioning products, specifically the estimation of quantity of various radionuclides, then using the spatial interpolation to predict contamination levels of unobserved locations. In their case, the Bayesian kriging model showed the most promising results, credited to the fact that it takes into account the parameter's uncertainty, however is criticized for being computationally expensive. While Wiskotten et al.'s study focuses on radiological characterization in decommissioning projects, our research centers on predicting crime occurrences in Vancouver using spatial modeling techniques. Our aim is to assess the robustness of the Bayesian kriging approach compared to the universal kriging approach. We will evaluate our models based on the estimates expected log predictive density, the LOO information criterion and root mean squared error (RMSE) of one such test set.

INITIAL EXPLORATION

Pre-Processing

Upon first analysis of the raw data imported from the Vancouver Police Department, the size and shape of the data made it difficult to work with. Many attempts were made to clean and work with the data as is, but to present an informative and meaningful analysis, we decided instead to alter the data so rather than being a set of points where each point represents one single crime reported, it is a map of equally spaced grid cells that contains a count of how many crimes occured. This was done using ArcGIS Pro, by creating a "fishnet" the size of vancouver with 500m x 500m grid cells and summarizing the number of crimes, then adding in the centre longitude (x-coord) and latitude (y-coord) for each grid, then exporting it as a CSV file.

Data Exploration

To better understand the data, some analysis was done to better understand the relationships & to inform choices made in our model. A summary of the number of crimes per month showed approximately equal numbers each 12 months, consistent with past research of crime distribution in areas with temperate conditions, and suggesting that a simpler model that doesn't account for spatiotemporal relationships could still yield promising results. (Linning, 2015)

To quantify the spatial autocorrelation within the dataset, the Global Moran's Index was calculated. A positive Moran's Index suggests that nearby or neighbouring locations have similar values (indicating positive spatial autocorrelation), while a negative value suggests that nearby locations have dissimilar values (negative spatial autocorrelation). A value close to zero indicates a random spatial pattern. This data set had a Moran's Index of 0.644, which on a scale from -1 to +1 suggests a positive spatial autocorrelation. This implies that we should be able to use the distance between the points to spatially interpolate missing data.

METHODOLOGY & DATA ANALYSIS

This study employed two kriging models with variations in their structure to attempt to find the best ways to spatially interpolate crime densities from the data set formed. They differ primarily in their handling of the spatial structure and the statistical assumptions about the underlying processes. To compare these two models, our original crime dataset is split 80/20 into training and testing sets, and a prediction is conducted for the counts of the testing set, which is repeated using leave-one-out cross-validation to compare for various splits. We will evaluate our models based on the estimates of expected log predictive density, the LOO information criterion and a sample root mean squared error (RMSE) on one of the tests.

Model Specifications

The data entered into the model includes `N`, the number of training points, `x`, `y`, the respective longitude and latitude of the training points, `crimes`, the count of crimes of each of the training points, `N_new`, the number of testing points, and `x_new`, `y_new`, the respective longitude and latitude of the training points. Model 1 leverages an ordinary kriging framework, characterized by the four parameters: `alpha` represents the baseline crime density, which is assumed to be constant across the spatial field, with the prior $\alpha \sim Normal(0, 10)$ `rho` defines the range parameter or length scale, which influences how quickly the correlation would decay with distance, with the prior $\rho \sim InvGamma(2, 1)$ `sigma` is the spatial variance, controlling the expected variability related to its spatial factors with the prior $\sigma \sim Normal(0, 1)$ `sigma_err` accounts for the random noise that may occur in the dataset, with the prior $\sigma_{err} \sim Normal(0, 1)$ The spatial structure is modeled using a Gaussian process with the covariance between any two points `i` and `j` are given by $dist_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, the Euclidean distance matrix computed from longitude and latitude data. The crimes are then assumed to follow a multivariate normal distribution $crimes \sim Multi-Normal(\mu, \sum)$, where $\sum = \sigma exp(\frac{-dist}{\rho}) + \sigma_{err} I$ and $\mu[i] = \alpha$, since we assume constant mean.

Model 2 simplifies the spatial structure by directly estimating a zero-mean multivariate normal distribution of the crime counts - it similarly uses a covariance matrix for the Gaussian function of distance, but utilizes a Chloseky decomposition for computational efficiency.

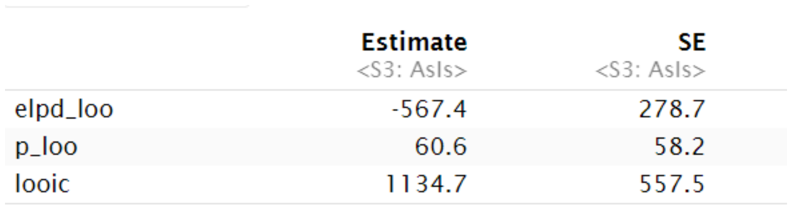
Prior Justification

In both models, the parameters are given the same priors for their equivalent variables to compare the nature of the models rather than the parameters themselves. An inverse gamma prior for `rho` was selected since we expect a level of spatial continuity, and the longer tail was to accommodate the possibility of significant spatial dependence, as the Global Moran's Index on this data showed high autocorrelation. The normal priors for `sigma` and `sigma_err` assume most variability near the mean, reflecting the uncertainty we had about our assumption. The broad normal prior `alpha` in Model 1 was to accommodate different baseline densities. While there is a distinct lack of use of kriging in other research related to the spatial interpolation of similar crime data, these priors (notably using normal priors for both sigma & sigma noise) are common choices in other fields when conducting kriging, and provided better results when tested. (Wiskotten et al., 2023; Brown & Brubaker, 2012; Watson, 2018)

Results

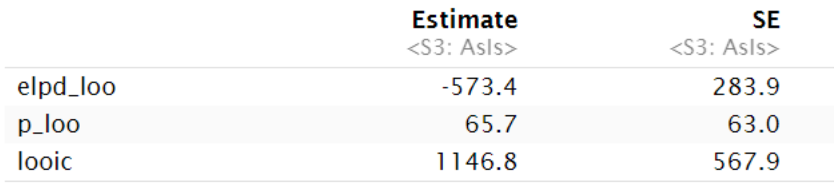
The results from the 'loo' function provide estimates for the expected log predictive density(ELPD), the effective number of parameters and the Leave-One-Out(LOO) information criterion. These metrics offer insights into each of two different model's predictive accuracy and complexity. We used these metrics to compare to Model 1 and Model 2, examining which method would yield better predictive performance.

We built two different Kriging models with Stan for both Model 1 and Model 2 to compare results: one using the original data (standard model) and another using the logarithmic transformation (log model). To determine which model have better predictive performance, we employed the 'loo_compare' function to evaluate the fit of the models.The decision between the standard model and the log model allows us to find the suitable model to compare two Kriging models.



	Estimate <S3: AsIs>	SE <S3: AsIs>
elpd_loo	-567.4	278.7
p_loo	60.6	58.2
looic	1134.7	557.5

Figure 1: Leave-One-Out Cross Validation Scores for Log Model 2



	Estimate <S3: AsIs>	SE <S3: AsIs>
elpd_loo	-573.4	283.9
p_loo	65.7	63.0
looic	1146.8	567.9

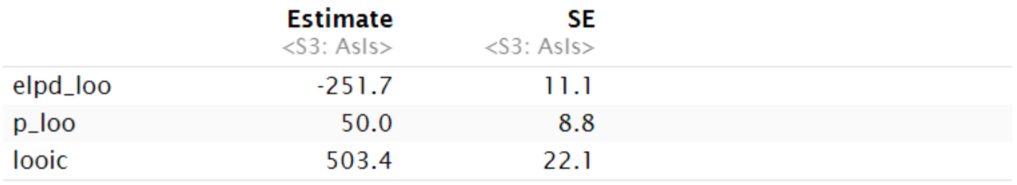
Figure 2: Leave-One-Out Cross Validation Scores for Log Model 2

The above figures shows that the results for Standard Model 2 and Log Model 2. The Expected Log Predictive Density (ELPD) is a measure of the model's predictive performance, as it serves as a theoretically motivated metric for evaluating predictive ability (Oelrich & Villani, 2024). A higher ELPD suggest better predictive accuracy. Our standard Model 2 had an ELPD estimate of -3039.1 with a standard error of 566.8, indicating some variability in the predictions. The log model using Model 2 showed a slightly better ELPD of -3028.9 with a standard error of 565.3, suggesting improved predictive performance.

The effective number of parameters is a measure of model complexity, indicating how much the model has adapted to the data, as discussed by Kheiri, Kimber, and Reza Meshkani (2007). In our standard Model 2, this value was 383.7 with a standard error of 112.2. The alternative log model had a slightly lower effective number of parameters at 372.9 with a standard error of 111.9, indicating that it might be less prone to overfitting.

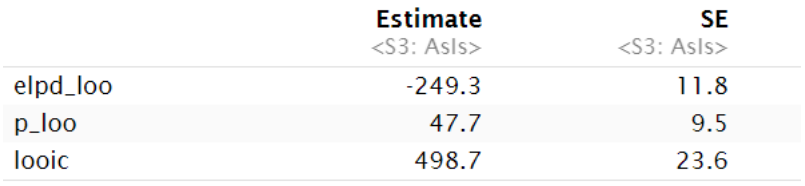
The LOO information criterion(LOO-IC) combines the ELPD and the effective number of parameters to estimate the expected predictive error. A lower LOO-IC value indicates better performance. The standard Model 2 produced a LOO-IC of 6078.3 with a standard error of 1133.6, while the log Model 2 yielded a LOO-IC of 6057.7 with a standard error of 1130.6. The lower LOO-IC in the log Model 2 suggests it has slightly better predictive performance.

These results indicate that the log model has slightly better predictive accuracy and lower complexity compared to the standard Bayesian Kriging model. However, the two model shows a very close performance comparison in Bayesian model.



	Estimate <S3: AsIs>	SE <S3: AsIs>
elpd_loo	-251.7	11.1
p_loo	50.0	8.8
looic	503.4	22.1

Figure 3: Leave-One-Out Cross Validation Scores for Model 1



	Estimate <S3: AsIs>	SE <S3: AsIs>
elpd_loo	-249.3	11.8
p_loo	47.7	9.5
looic	498.7	23.6

Figure 4: Leave-One-Out Cross Validation Scores for Log Model 1

It is clearer when we look at the results of the Universal Kriging model that the log model generally performs better than the standard model. The figures above shows the performance for each of Standard Model 1 and Log Model 1. Therefore, to further evaluate the models, we plan to compare the Universal Kriging Model and Bayesian Kriging Model using the log transformation to see which approach provides more robust predictions.

As the figures above indicates, our Model 2 (note that these are now referring to the log models) has an ELPD estimate of -3028.9 with a standard error of 565.3, indicating some variability in the predictions. The Model 1 has an lower ELPD estimate of -427.1 with a smaller standard error of 31.3. This indicates that the Model 1 may have better predictive performance compared to Model 2, as it shows a higher ELPD estimate and a lower standard error, signifying potentially more reliable predictions with less variability.

The Model 2 had an effective number of parameters was 372.9 with a standard error of 111.9. The Model 1 had a lower effective number of parameters at 204.0 with a standard error of 25.4. It indicates that the Model 1 is less complex and potentially less prone to overfitting.

The Model 2 produced a LOO-IC of 6057.7 with a standard error of 1130.6 while the Model 1 had a lower LOO-IC of 854.1 with a smaller standard error of 62.6. This indicates that the Model 1 performs better in terms of predictive performance.

Conclusion

In conclusion, our study demonstrates a comparative analysis of two different Kriging models for crime density prediction in the city of Vancouver, revealing the predictive capabilities of each approach. We found that the our first model which most closely represents Universal Kriging outperformed our second Kriging model which most closely represents a standard Bayesian kriging model in our study based on the metrics found in leave on out cross validation, although the exact reason for this performance difference requires further investigation.

One significant limitation of our study is the assumption of stationary in spatial processes. Real-word scenarios often exhibit non-stationary behavior, where mean, variance, and autocorrelation structures may vary over time or space. Additionally, the choice of priors and hyperparameters can influence model performance and generalization ability. Exploring alternative Kriging models such as ordinary Kriging, or optimizing model parameters could potentially yield improved results. Another potential limitation was the lack of other variables considered in our analysis. Mentioned previously, no temporal input was included into our interpolation models based on the preliminary analysis that showed crime density was relatively even in each month. Future research could include other factors such as reported time of day or even aspatial and non-temporal factors such as type of crime could have created a better informed hierarchical model, however were omitted due to its complexity in novel spatial interpolation analysis. It is also important to consider that the data were are using is sourced solely from crime reports, which have been known to contain biases based on location and other factors and may not fully represent crime occurrences. (Westin, 2021)

Our study holds significance as it bridges the gap between theoretical modeling techniques and practical applications in crime prediction, offering insights into the effectiveness of Bayesian and Universal Kriging models. Our findings underscore the importance of robust spatial modeling approaches in addressing complex real-world challenges, emphasizing the need for informed decision-making in crime management and urban development initiatives.

Contributions of each team member:

Choice Data/Data preprocessing: Zela, Youjung

Universal Kriging Model: Zela

Bayesian Kriging model: Zela

LOO-CV Model Adaptations: Youjung

Log Model Adaptations: Youjung

Model Fitting: Zela

Results Interpretation: Zela(rmse, plots), Youjung(loo)

Intro/background/results: Youjung

Methodology/Exploration: Zela

Conclusion:Zela, Youjung

Appendix A

```r
library(ggplot2)
library(sf)
```

```
## Linking to GEOS 3.11.2, GDAL 3.6.2, PROJ 9.2.0; sf_use_s2() is TRUE
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(rstan)
```

```
## Loading required package: StanHeaders

## Warning: package 'StanHeaders' was built under R version 4.3.2

##
## rstan version 2.32.5 (Stan version 2.32.2)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)

## Do not specify '-march=native' in 'LOCAL_CPPFLAGS' or a Makevars file
```

```
library(tidybayes)
```

```
## Warning: package 'tidybayes' was built under R version 4.3.2
```

```
library(magrittr)
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:rstan':
##
##     extract
```

```
library(loo)
```

```
## Warning: package 'loo' was built under R version 4.3.2

## This is loo version 2.6.0

## - Online documentation and vignettes at mc-stan.org/loo

## - As of v2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use the 'cores' ar

## - Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see https://gi

##
## Attaching package: 'loo'

## The following object is masked from 'package:rstan':
##
##     loo
```

```
#all the data things
set.seed(123)  # For reproducibility

crime_data <- read.csv("CrimeDensity2023fixed2.csv")
crime_data <- crime_data[-c(201:nrow(crime_data)),] #for now
crime_data$COUNT <- crime_data$Point_Count
head(crime_data)
```

```
##   Shape_Length Shape_Area Point_Count X_Coord  Y_Coord COUNT
## 1         2000     250000           4 1208982 469299.3     4
## 2         2000     250000           2 1209482 469299.3     2
## 3         2000     250000           7 1207982 469799.3     7
## 4         2000     250000          26 1208482 469799.3    26
## 5         2000     250000          50 1208982 469799.3    50
## 6         2000     250000          30 1209482 469799.3    30
```

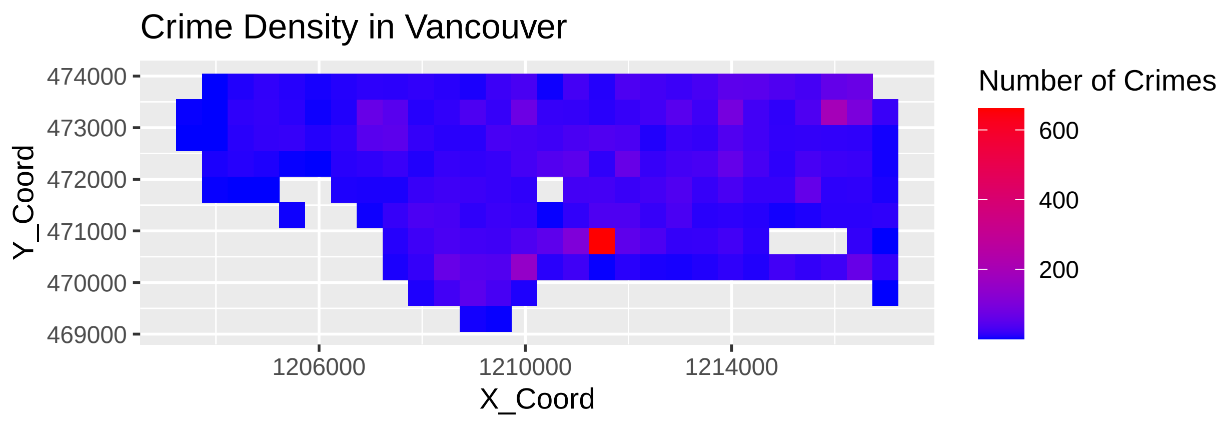
```
library(readr)
#crime_data <- read_csv("~/Downloads/crime_records.csv")
```

```
# missing data exmination
sum(is.na(crime_data$NEIGHBOURHOOD))
```

```
## [1] 0
```

```
# crime counts by year
# ggplot(data = crime_data, aes(x = YEAR)) +
#   geom_bar() +
#   facet_wrap(~NEIGHBOURHOOD, ncol = 5)
```

```
# Assuming crime_data is your dataframe
ggplot(crime_data, aes(x = X_Coord, y = Y_Coord, fill = COUNT)) +
  geom_tile() +  # creates the grid cells
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "Crime Density in Vancouver", fill = "Number of Crimes")+
  coord_fixed(ratio = 1)  # keeps the aspect ratio of 1:1
```



```r
# Set seed for reproducibility
set.seed(123)

# Sample 80% of the data to keep
train_indices <- sample(nrow(crime_data), size = 0.8 * nrow(crime_data))
train_data <- crime_data[train_indices, ]
test_data <- crime_data[-train_indices, ]
```

Spatial Model

```stan
data {
  int<lower=0> N; // number of data points (train)
  vector[N] x; // long of train
  vector[N] y; // lat of train
  vector[N] crimes; // count of train

  int<lower=0> N_new; // number of data points (test)
  vector[N_new] x_new; // long of test
  vector[N_new] y_new; //lat of test
}

parameters {
  real alpha; //intercept
  real<lower=0> rho; //range
  real<lower=0> sigma; //spatial variance
  real<lower=0> sigma_err; // measurement error
}

model {
  vector[N] mu;

  for (i in 1:N) {
    mu[i] = alpha;
  }

  // Priors
  alpha ~ normal(0, 10);
  rho ~ inv_gamma(5, 5);
  sigma ~ normal(0, 1);
  sigma_err ~ normal(0, 1);

  // Spatial structure
  {
    matrix[N, N] dist;
    for (i in 1:N) {
      for (j in 1:N) {
        dist[i, j] = sqrt(square(x[i] - x[j]) + square(y[i] - y[j]));
      }
    }
    crimes ~ multi_normal(mu, sigma * exp(-dist/rho) + diag_matrix(rep_vector(sigma_err, N)));
  }
}

generated quantities{
  vector[N_new] pred_crimes; //predicted crimes
  matrix[N, N_new] new_dist; //distance from old to new
  for (i in 1:N){
    for (j in 1:N_new){
      new_dist[i, j] = exp(-sqrt(square(x[i] - x_new[j]) + square(y[i] - y_new[j])) / rho);
    }
  }

  for (i in 1:N_new){
    pred_crimes[i] = normal_rng(alpha + dot_product(new_dist[, i], crimes - rep_vector(alpha, N)), sigma
  }
}
```

Spatial Model with loocv

```
data {
  int<lower=0> N; // number of data points (train)
  vector[N] x; // long of train
  vector[N] y; // lat of train
  vector[N] crimes; // count of train

  int<lower=0> N_new; // number of data points (test)
  vector[N_new] x_new; // long of test
  vector[N_new] y_new; //lat of test
}

parameters {
  real alpha; //intercept
  real<lower=0> rho; //range
  real<lower=0> sigma; //spatial variance
  real<lower=0> sigma_err; // measurement error
}

model {
  vector[N] mu;

  for (i in 1:N) {
    mu[i] = alpha;
  }

  // Priors
  alpha ~ normal(0, 10);
  rho ~ inv_gamma(5, 5);
  sigma ~ normal(0, 1);
  sigma_err ~ normal(0, 1);

  // Spatial structure
  {
    matrix[N, N] dist;
    for (i in 1:N) {
      for (j in 1:N) {
        dist[i, j] = sqrt(square(x[i] - x[j]) + square(y[i] - y[j]));
      }
    }
    crimes ~ multi_normal(mu, sigma * exp(-dist/rho) + diag_matrix(rep_vector(sigma_err, N)));
  }
}

generated quantities{
  vector[N_new] log_lik;
  vector[N_new] pred_crimes; //predicted crimes
  matrix[N, N_new] new_dist; //distance from old to new
  for (i in 1:N){
    for (j in 1:N_new){
      new_dist[i, j] = exp(-sqrt(square(x[i] - x_new[j]) + square(y[i] - y_new[j])) / rho);
    }
  }

  for (i in 1:N_new){
    pred_crimes[i] = normal_rng(alpha + dot_product(new_dist[, i], crimes - rep_vector(alpha, N)), sigma
    log_lik[i] = normal_lpdf(crimes[i] | pred_crimes[i], sigma_err);
  }
}
```

Fit Spatial Model with loocv

```
standata <- list(N = nrow(train_data),
                         x = train_data$X_Coord,
                         y = train_data$Y_Coord,
                         crimes = train_data$COUNT,
                         N_new = nrow(test_data),
                         x_new = test_data$X_Coord,
                         y_new = test_data$Y_Coord)
fit = sampling(geo_model_loo,
               data = standata,
               chains = 1,
               refresh = 0,
               iter = 2000)
```

```
log_lik_1 <- extract_log_lik(fit, merge_chains = FALSE)
r_eff <- relative_eff(exp(log_lik_1), cores = 2)
loo_1 <- loo(log_lik_1, r_eff, cores = 2)
```

```
## Warning: Relative effective sample sizes ('r_eff' argument) not specified.
## For models fit with MCMC, the reported PSIS effective sample sizes and
## MCSE estimates will be over-optimistic.

## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```
print(loo_1)
```

```
##
## Computed from 1000 by 40 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -251.7 11.1
## p_loo        50.0  8.8
## looic       503.4 22.1
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)      0     0.0%   <NA>
##  (0.5, 0.7]   (ok)        6    15.0%   65
##    (0.7, 1]   (bad)      19    47.5%   21
##    (1, Inf)   (very bad) 15    37.5%   2
## See help('pareto-k-diagnostic') for details.
```

```
# Compare this model to log matrix
standata_2 <- standata
standata_2$x <- log(standata$x)
standata_2$y <- log(standata$y)
standata_2$x_new <- log(standata$x_new)
standata_2$y_new <- log(standata$y_new)
standata_2$crimes <- log(standata$crimes)
```

```r
fit_2 <- stan(fit = fit,
              data = standata,
              chains = 1,
              refresh = 0)
log_lik_2 <- extract_log_lik(fit_2, merge_chains = FALSE)
r_eff_2 <- relative_eff(exp(log_lik_2))
loo_2 <- loo(log_lik_2, r_eff = r_eff_2, cores = 2)
```

```
## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```r
print(loo_2)
```

```
##
## Computed from 1000 by 40 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -249.3 11.8
## p_loo        47.7  9.5
## looic       498.7 23.6
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)      1     2.5%   118
##  (0.5, 0.7]   (ok)        4    10.0%   87
##    (0.7, 1]   (bad)      26    65.0%   14
##    (1, Inf)   (very bad)  9    22.5%   2
## See help('pareto-k-diagnostic') for details.
```

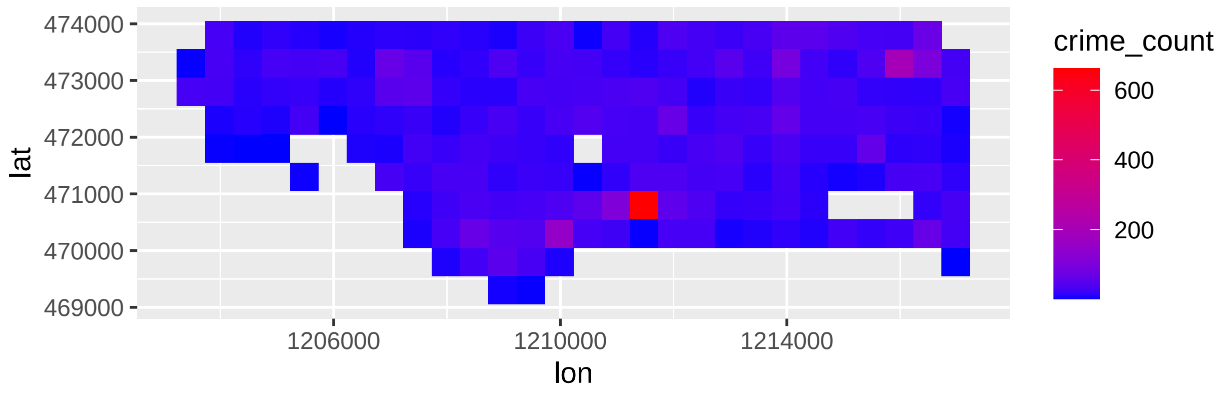
```r
diff <- loo_compare(loo_1, loo_2)
print(diff)
```

```
##        elpd_diff se_diff
## model2  0.0       0.0
## model1 -2.4       3.0
```

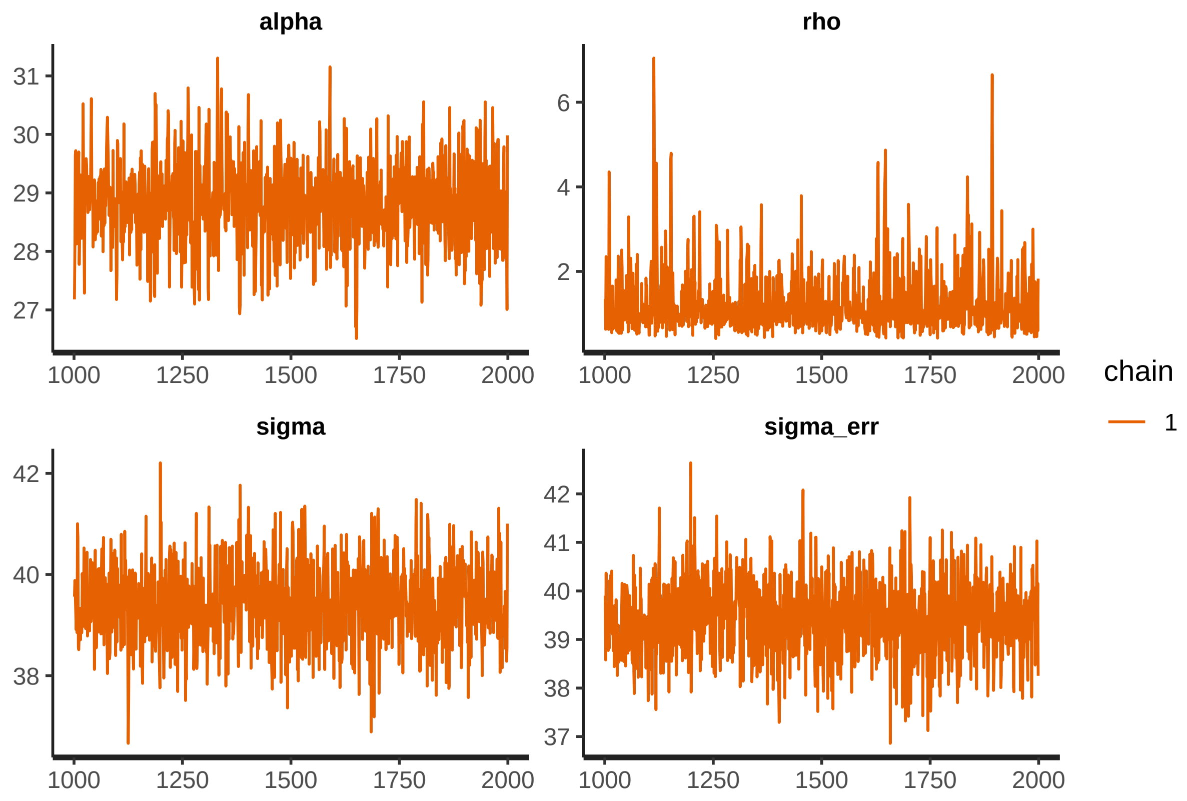
```r
samples = rstan::extract(fit)

predicted_crimes <- samples$pred_crimes

mean_predicted_crimes <- apply(predicted_crimes, 2, mean)

predicted_data <- data.frame(lon = test_data$X_Coord,
                             lat = test_data$Y_Coord,
                             crime_count = mean_predicted_crimes)
```

```r
train <- data.frame(
  lon = train_data$X_Coord,
  lat = train_data$Y_Coord,
  crime_count = train_data$COUNT
)

# Combine both datasets
full_data <- rbind(
  train[, c("lon", "lat", "crime_count")],
  predicted_data[, c("lon", "lat", "crime_count")]
)

full_data <- na.omit(full_data)
```

```r
ggplot(full_data, aes(x = lon, y = lat, fill = crime_count)) +
  geom_tile() +
  scale_fill_gradient(low = "blue", high = "red")+
  coord_fixed(ratio = 1)
```



```r
traceplot(fit, pars = c("alpha", "rho", "sigma", "sigma_err"))
```



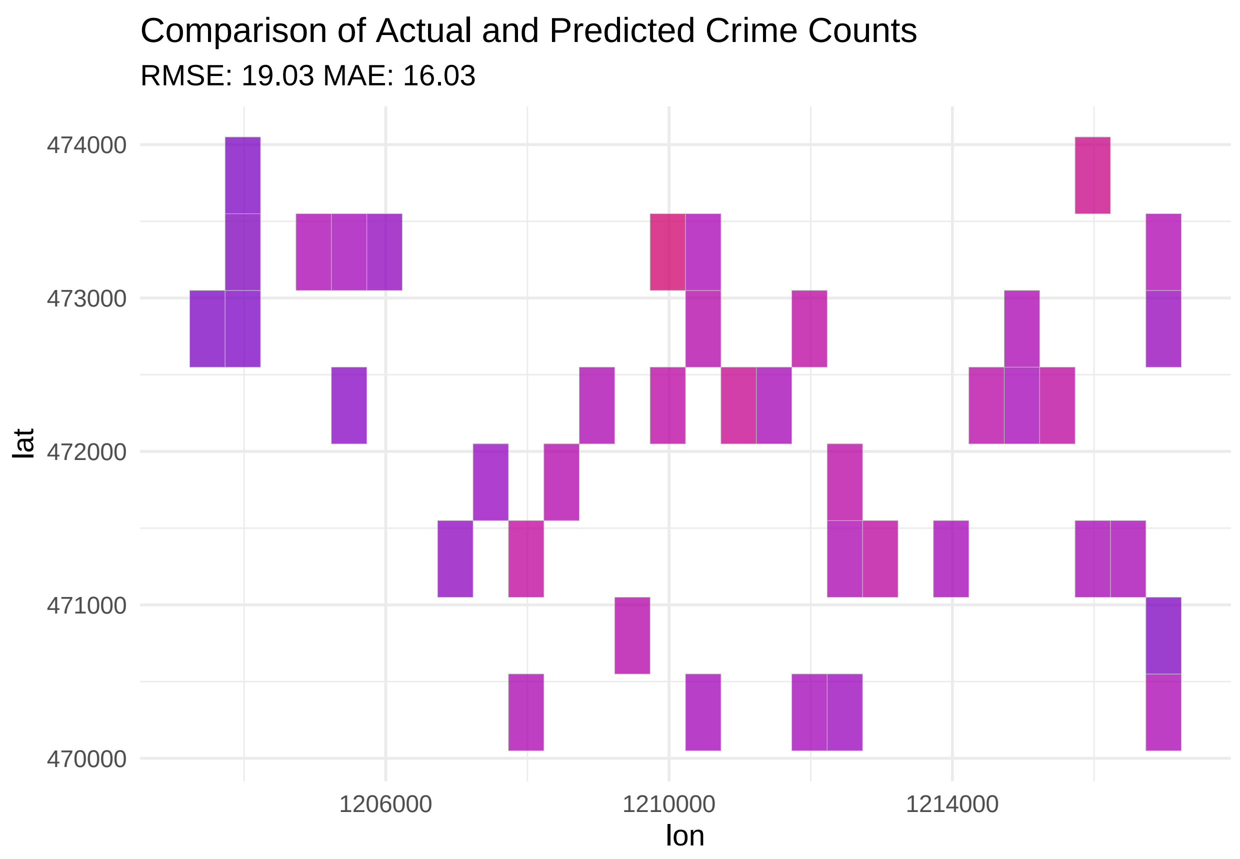
```
# Calculate RMSE
rmse <- sqrt(mean((predicted_data$crime_count - test_data$COUNT)^2))
# Calculate MAE
mae <- mean(abs(predicted_data$crime_count - test_data$COUNT))

# Print the metrics
print(paste("Root Mean Squared Error:", rmse))
```

```
## [1] "Root Mean Squared Error: 19.0262825432858"
```

```
print(paste("Mean Absolute Error:", mae))
```

```
## [1] "Mean Absolute Error: 16.0345301615771"
```

```
# Combine test data and predicted data for plotting
comparison_data <- cbind(test_data, predicted_data)

ggplot(comparison_data, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = COUNT), alpha = 0.5) +  # Actual counts
  geom_tile(aes(fill = crime_count), color = "grey", alpha = 0.5) +  # Predicted counts
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "Comparison of Actual and Predicted Crime Counts",
       subtitle = paste("RMSE:", round(rmse, 2), "MAE:", round(mae, 2))) +
  theme_minimal() +
  theme(legend.position = "none")
```



Bayesian Spatial Model with loocv

```
data {
  int<lower=0> N; // number of data points (train)
  vector[N] x; // longitude of train data
  vector[N] y; // latitude of train data
  vector[N] crimes; // crime count of train data

  int<lower=0> N_new; // number of data points (test)
  vector[N_new] x_new; // longitude of test data
  vector[N_new] y_new; // latitude of test data
}

parameters {
  real<lower=0> rho;
  real<lower=0> sigma; // scale of the output
  real<lower=0> sigma_err; // noise level
}

transformed parameters {
  matrix[N, N] covar_matrix;
  for (i in 1:N) {
    for (j in i:N) {
      real dist = sqrt(square(x[i] - x[j]) + square(y[i] - y[j]));
      covar_matrix[i, j] = sigma * exp(-dist / rho);
      if (i != j) {
        covar_matrix[j, i] = covar_matrix[i, j]; // Symmetric matrix
      }
    }
    covar_matrix[i, i] += sigma_err^2; // adding noise variance to the diagonal
  }
}

model {
  // Priors
  rho ~ inv_gamma(5, 5);
  sigma ~ normal(0, 1);
  sigma_err ~ normal(0, 0.1);

  // Likelihood
  crimes ~ multi_normal_cholesky(rep_vector(0, N), cholesky_decompose(covar_matrix));
}

generated quantities {
  vector[N_new] log_lik;
  vector[N_new] pred_crimes; // predicted crimes
  matrix[N, N_new] new_dist; // distance from old to new

  // Compute distances for the new matrix
  for (i in 1:N) {
    for (j in 1:N_new) {
      new_dist[i, j] = sqrt(square(x[i] - x_new[j]) + square(y[i] - y_new[j]));
    }
  }

  // Covariance matrix between train and test data
  matrix[N, N_new] k_x_xnew;
  for (i in 1:N) {
    for (j in 1:N_new) {
      k_x_xnew[i, j] = sigma * exp(-new_dist[i, j] / rho);
    }
  }

  // Prediction mean
  matrix[N_new, N] k_xnew_x = k_x_xnew'; // Transpose
  vector[N] alpha = cholesky_decompose(covar_matrix) \ crimes; // Cholesky factorization and solve
  pred_crimes = k_xnew_x * alpha + mean(crimes);

  // cross val
  for (i in 1:N_new){
    log_lik[i] = normal_lpdf(crimes[i] | pred_crimes[i], sigma_err);
  }
}
```

Fit Bayesian Spatial Model with loocv

```
standata <- list(N = nrow(train_data),
                         x = train_data$X_Coord,
                         y = train_data$Y_Coord,
                         crimes = train_data$COUNT,
                         N_new = nrow(test_data),
                         x_new = test_data$X_Coord,
                         y_new = test_data$Y_Coord)
fit = sampling(bayes_geo_model_loo,
               data = standata,
               chains = 1,
               refresh = 0,
               iter = 2000)
```

```
log_lik_1 <- extract_log_lik(fit, merge_chains = FALSE)
r_eff <- relative_eff(exp(log_lik_1), cores = 2)
loo_1 <- loo(log_lik_1, r_eff, cores = 2)
```

```
## Warning: Relative effective sample sizes ('r_eff' argument) not specified.
## For models fit with MCMC, the reported PSIS effective sample sizes and
## MCSE estimates will be over-optimistic.

## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```
print(loo_1)
```

```
##
## Computed from 1000 by 40 log-likelihood matrix
##
##          Estimate    SE
## elpd_loo   -567.4 278.7
## p_loo        60.6  58.2
## looic      1134.7 557.5
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     39    97.5%   756
##  (0.5, 0.7]   (ok)        0     0.0%   <NA>
##    (0.7, 1]   (bad)       0     0.0%   <NA>
##    (1, Inf)   (very bad)  1     2.5%   2
## See help('pareto-k-diagnostic') for details.
```

```
# Compare this model to log matrix
standata_2 <- standata
standata_2$x <- log(standata$x)
standata_2$y <- log(standata$y)
standata_2$x_new <- log(standata$x_new)
standata_2$y_new <- log(standata$y_new)
standata_2$crimes <- log(standata$crimes)
```

```r
fit_2 <- stan(fit = fit,
              data = standata,
              chains = 1,
              refresh = 0)
log_lik_2 <- extract_log_lik(fit_2, merge_chains = FALSE)
r_eff_2 <- relative_eff(exp(log_lik_2))
loo_2 <- loo(log_lik_2, r_eff = r_eff_2, cores = 2)
```

```
## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```r
print(loo_2)
```

```
##
## Computed from 1000 by 40 log-likelihood matrix
##
##          Estimate    SE
## elpd_loo   -573.4 283.9
## p_loo        65.7  63.0
## looic      1146.8 567.9
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     39    97.5%   442
##  (0.5, 0.7]   (ok)        0     0.0%   <NA>
##    (0.7, 1]   (bad)       0     0.0%   <NA>
##    (1, Inf)   (very bad)  1     2.5%   1
## See help('pareto-k-diagnostic') for details.
```

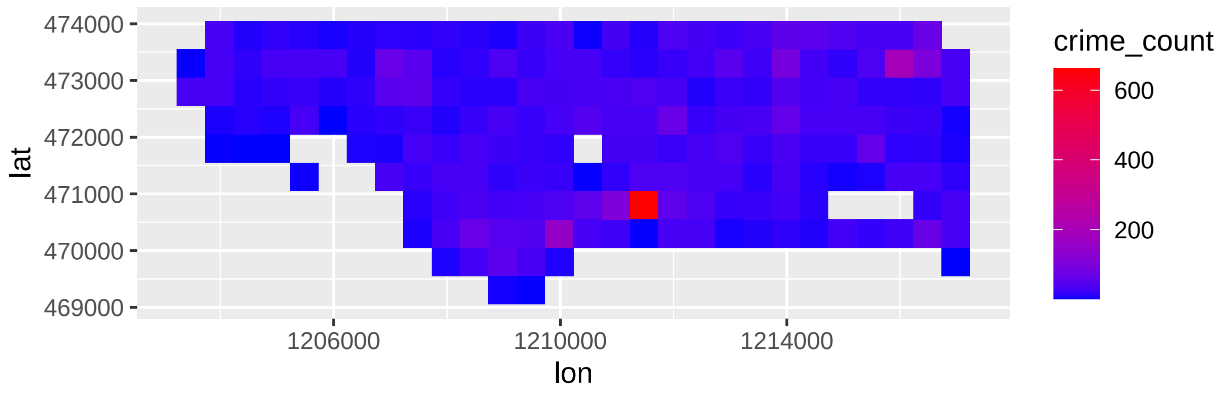
```r
diff <- loo_compare(loo_1, loo_2)
print(diff)
```

```
##        elpd_diff se_diff
## model1  0.0       0.0
## model2 -6.1       5.2
```

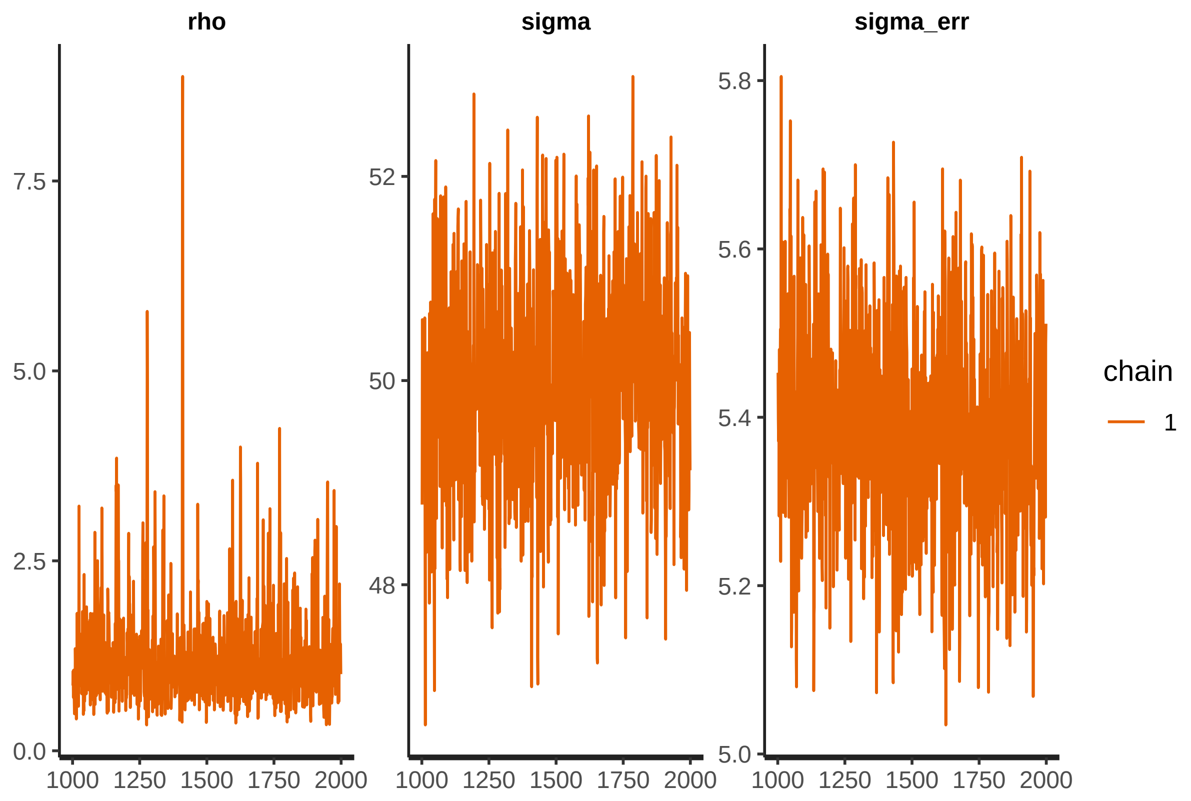
```r
samples = rstan::extract(fit)

predicted_crimes <- samples$pred_crimes

mean_predicted_crimes <- apply(predicted_crimes, 2, mean)

predicted_data <- data.frame(lon = test_data$X_Coord,
                             lat = test_data$Y_Coord,
                             crime_count = mean_predicted_crimes)
```

```r
train <- data.frame(
  lon = train_data$X_Coord,
  lat = train_data$Y_Coord,
  crime_count = train_data$COUNT
)

# Combine both datasets
full_data <- rbind(
  train[, c("lon", "lat", "crime_count")],
  predicted_data[, c("lon", "lat", "crime_count")]
)

full_data <- na.omit(full_data)
```

```r
ggplot(full_data, aes(x = lon, y = lat, fill = crime_count)) +
  geom_tile() +
  scale_fill_gradient(low = "blue", high = "red")+
  coord_fixed(ratio = 1)
```



```r
traceplot(fit, pars = c("rho", "sigma", "sigma_err"))
```



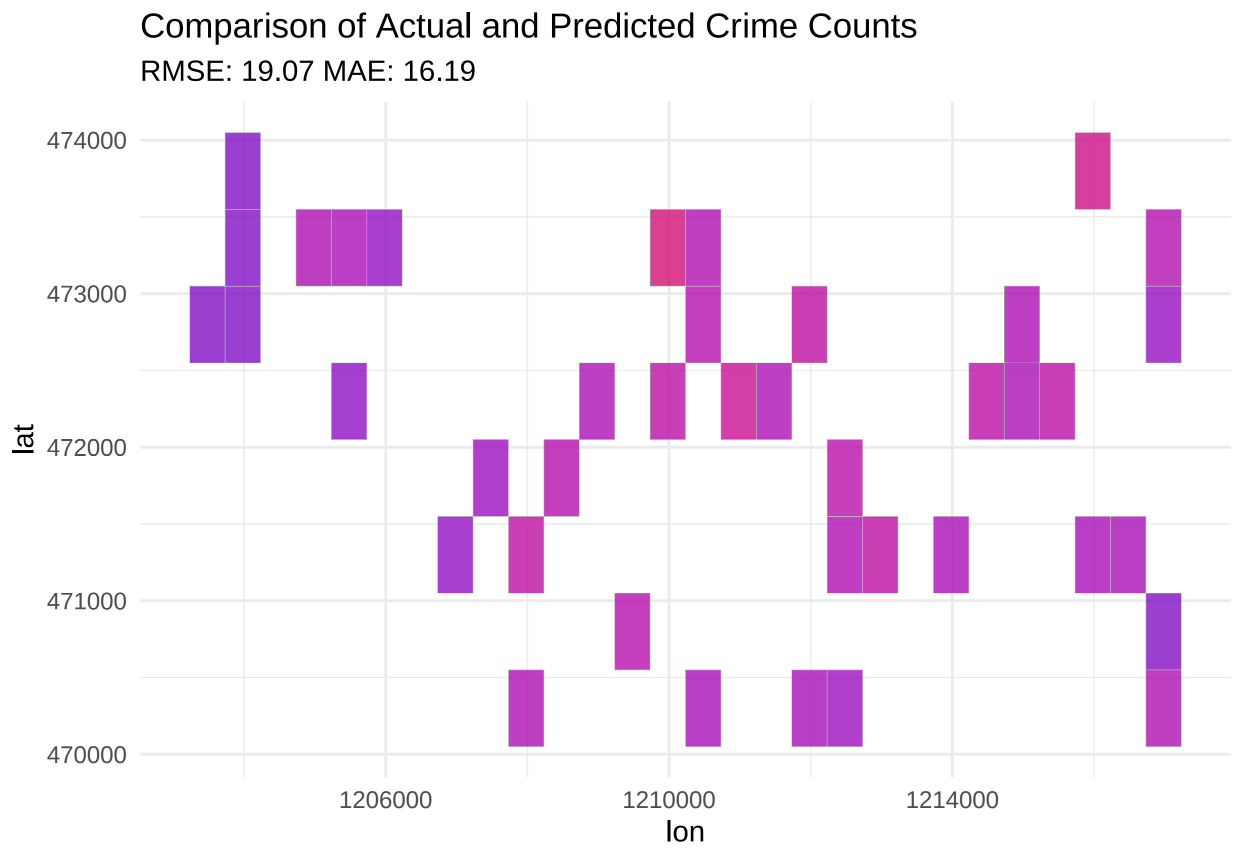
```r
# Calculate RMSE
rmse <- sqrt(mean((predicted_data$crime_count - test_data$COUNT)^2))
# Calculate MAE
mae <- mean(abs(predicted_data$crime_count - test_data$COUNT))

# Print the metrics
print(paste("Root Mean Squared Error:", rmse))
```

```
## [1] "Root Mean Squared Error: 19.0657507474135"
```

```r
print(paste("Mean Absolute Error:", mae))
```

```
## [1] "Mean Absolute Error: 16.1896875"
```

```r
# Combine test data and predicted data for plotting
comparison_data <- cbind(test_data, predicted_data)

ggplot(comparison_data, aes(x = lon, y = lat)) +
  geom_tile(aes(fill = COUNT), alpha = 0.5) +  # Actual counts
  geom_tile(aes(fill = crime_count), color = "grey", alpha = 0.5) +  # Predicted counts
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "Comparison of Actual and Predicted Crime Counts",
       subtitle = paste("RMSE:", round(rmse, 2), "MAE:", round(mae, 2))) +
  theme_minimal() +
  theme(legend.position = "none")
```



References

Andresen, M.A. (2006), A spatial analysis of crime in Vancouver, British Columbia: a synthesis of social disorganization and routine activity theory. The Canadian Geographer / Le Géographe canadien, 50: 487-502. https://doi.org/10.1111/j.1541-0064.2006.00159.x

Brown, E., & Brubaker, M. (2012). Kriging model. Stan Users Group Discussion - Kriging Model JAGS. https://groups.google.com/g/stan-users/c/PNRoDMQvzrs

Government Of Canada, S. C. (2023). Table 4 police-reported crime severity index and crime rate, by census metropolitan area, 2022 . Police-reported Crime Severity Index and crime rate, by census metropolitan area, 2022. https://www150.statcan.gc.ca/n1/daily-quotidien/230727/t004b-eng.htm

Kheiri, S., Kimber, A., & Reza Meshkani, M. (2007). Bayesian analysis of an inverse gaussian correlated frailty model. Computational Statistics & Data Analysis, 51(11), 5317–5326. https://doi.org/10.1016/j.csda.2006.09.026

Mooney, S. J., Bader, M. D., Lovasi, G. S., Neckerman, K. M., Rundle, A. G., & Teitler, J. O. (2018). Using universal kriging to improve neighborhood physical disorder measurement. Sociological Methods & Research, 49(4), 1163–1185. https://doi.org/10.1177/0049124118769103

Oelrich, O., & Villani, M. (2024, February 3). Modeling local predictive ability using power-transformed gaussian processes. arXiv.org. https://arxiv.org/abs/2402.02068

Sam, W. (2018, July 13). Geostatistical modelling with R and Stan. The Academic Health Economists' Blog. https://aheblog.com/2016/12/07/geostatistical-modelling-with-r-and-stan/

Vancouver Police Department (VPD) crime data. (2024). Vancouver Police Department (VPD) crime data. geoDASH Open Data. https://geodash.vpd.ca/opendata/

Watson, S. (2018, July 13). Geostatistical modelling with R and Stan. The Academic Health Economists' Blog. https://aheblog.com/2016/12/07/geostatistical-modelling-with-r-and-stan/

Wieskotten, M., Crozet, M., Iooss, B., Lacaux, C., & Marrel, A. (2023). A comparison between Bayesian and ordinary Kriging based on validation criteria: Application to radiological characterisation. Mathematical Geosciences, 56(1), 143–168. https://doi.org/10.1007/s11004-023-10072-y

Westin, M. (2023). Reported crime statistics in Toronto can be misleading. Reported Crime Statistics in Toronto can be Misleading. https://tellingstorieswithdata.com/inputs/pdfs/paper_one-2021-Morgaine_Westin.pdf

Zhang, Z., & Du, Q. (2019). A Bayesian Kriging regression method to estimate air temperature using remote sensing data. Remote Sensing, 11(7), 767. https://doi.org/10.3390/rs11070767