Monte Carlo Regional Kernel

Adam Howes

May 2020

Given k_s (kernel on locations e.g. Matern), we outline some possible methods for producing a suitable $k_{\mathcal{R}}$.

(Define `matern`, a version of the Matern kernel taking `r` a distance between points as input)

```
matern <- function(r, l = 2, nu = 1.5){
  if(!nu %in% c(1.5, 2.5)){
     errorCondition("Choose either nu = 1.5 or 2.5")
   }
  ifelse(nu == 1.5, (1 + sqrt(3)*r/l) * exp(-sqrt(3) * r/l),
                    (1 + sqrt(5)*r/l + 5*r^2/(3*l^2)) * exp(-sqrt(5) * r/l))
}
```

Centroid kernel One simple approach is based upon finding a point location which is somehow representative of the whole region. For example, let $c_i \in \mathcal{R}_i$ be the centroids of each region (with coordinates given by the arithmetic mean in each dimension.) The centroid kernel is given by $k_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j) = k_s(c_i, c_j)$.

Integrated kernel A more natural approach is to integrate the location based kernel over the regions of interest giving

$$k_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j) = \int_{\mathcal{R}_i} \int_{\mathcal{R}_j} k_s(s^{(i)}, s^{(j)}) ds^{(i)} ds^{(j)}. \tag{1}$$

Equation 1 directly takes into account all of the locations in each region, in constrast to the centroid approach. Although it is usually not possible to calculate this integral directly, supposing that samples may be drawn uniformly from each region, it may be approximated using random samples as follows. Suppose we have n collections of L_i samples drawn uniformly from inside each region $\mathcal{R}_i$

$$s_l^{(i)} \sim \mathcal{U}(\mathcal{R}_i), \quad l = 1, \dots, L. \tag{2}$$

Then a Monte Carlo estimate of $k_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j)$ is given by

$$k_{\mathcal{R}}(\mathcal{R}_i, \mathcal{R}_j) \approx \frac{1}{L_i L_j} \sum_{l=1}^{L_i} \sum_{m=1}^{L_j} k_s(s_l^{(i)}, s_m^{(j)}), \tag{3}$$

```
library(sf)
library(tidyverse)
library(reshape2)
library(viridis)
```

Import data:

```r
mw <- readRDS("~/Documents/phd/sae/data/all.rds") %>%
  filter(survey_id == "MW2015DHS")
```

Set $L_i = L$ (same sames for all regions) and consider $L = 1, 10, 100$:

```r
# Number of regions of Malawi
n <- nrow(mw)

# Sample sizes for each of the three experiments
sizes <- list(rep(1, n), rep(10, n), rep(100, n))

# Collect all samples
samples <- lapply(sizes, st_sample, x = mw)

# Function to produce covariance matrix from samples
sample_based_covariance <- function(sample) {

  # Distance between each point (all nsim * n of them)
  D <- st_distance(sample, sample)

  nsim <- length(sample) / n
  cov <- matrix(nrow = n, ncol = n)
  for(i in 1:n) {
    for(j in 1:n) {
      i_range <- ((i - 1) * nsim + 1):(i * nsim)
      j_range <- ((j - 1) * nsim + 1):(j * nsim)
      relevant_sample <- D[i_range, j_range]
      d <- mean(relevant_sample)
      cov[i, j] <- matern(d, l = 2, nu = 1.5)
    }
  }
  return(cov)
}

# Use the function just created
covs <- lapply(samples, sample_based_covariance)
```

Include centroids as a fourth experiment:

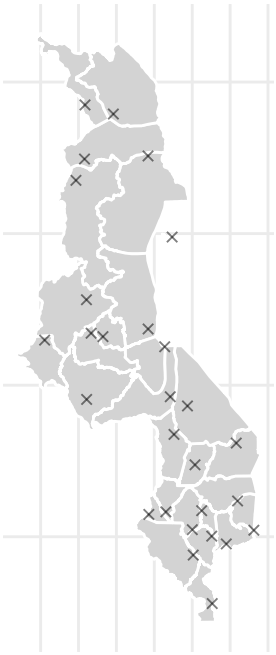
```r
samples[[4]] <- st_centroid(mw)
```

```
## Warning in st_centroid.sf(mw): st_centroid assumes attributes are constant over
## geometries of x
```

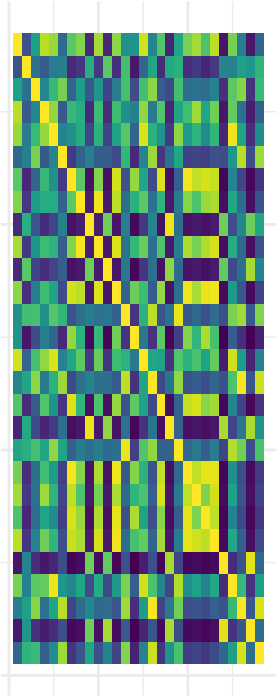
```r
D_cent <- st_distance(samples[[4]], samples[[4]])
cov_cent <- apply(D_cent, c(1, 2), matern)
covs[[4]] <- cov_cent
```

```r
# Undo 90 degree counter-clockwise rotation
rotate <- function(x) t(apply(x, 2, rev))

plot_experiment <- function(i, title) {

  sample_plot <- ggplot(mw) +
    geom_sf(fill = "lightgrey", color = "white") +
    geom_sf(data = samples[[i]], alpha = 0.5, shape = 4) +
    labs(x = "Longitude", y = "Latitude") +
    theme_minimal() +
    labs(fill = "",
         title = paste0(title),
         caption = "Sampled points") +
    theme(axis.title = element_blank(),
          axis.text = element_blank(),
          axis.ticks = element_blank())

  matrix_plot <- melt(rotate(covs[[i]])) %>%
    ggplot(aes(x = Var1, y = Var2, fill = value)) +
      geom_tile() +
      scale_fill_viridis() +
      theme_minimal() +
      labs(fill = "", caption = "Regional similarity matrix") +
      theme(axis.title = element_blank(),
            axis.text = element_blank(),
            axis.ticks = element_blank(),
            legend.position = "none")

  cowplot::plot_grid(sample_plot, matrix_plot)
}

plot1 <- plot_experiment(1, "1 point per region")
plot2 <- plot_experiment(2, "10 points per region")
plot3 <- plot_experiment(3, "100 points per region")
plot4 <- plot_experiment(4, "Centroids")

cowplot::plot_grid(plot1, plot2, plot3, plot4, ncol = 2)
```

1 point per region

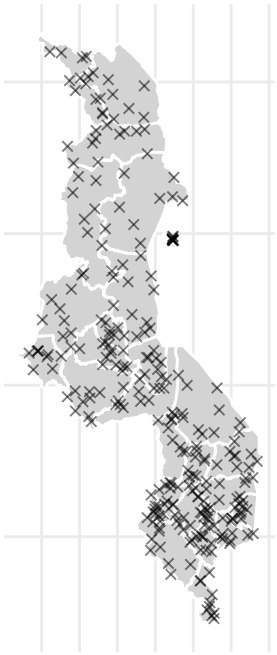


Sampled points

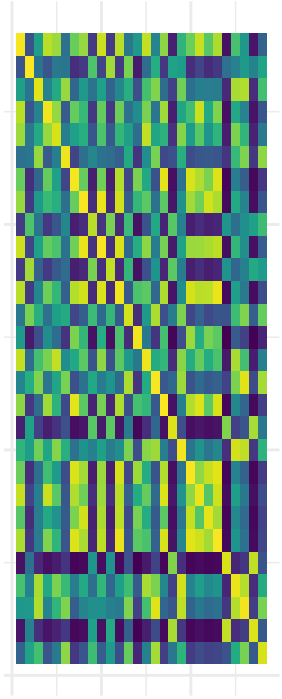


Regional similarity matrix

10 points per region

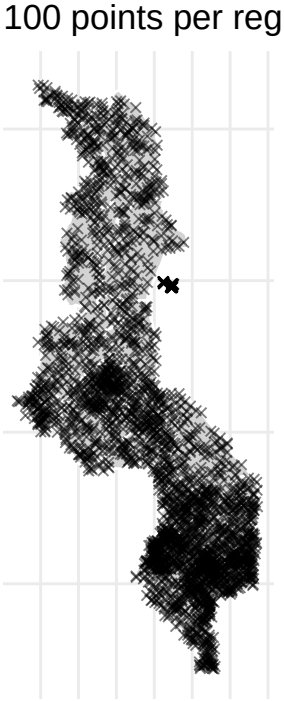


Sampled points

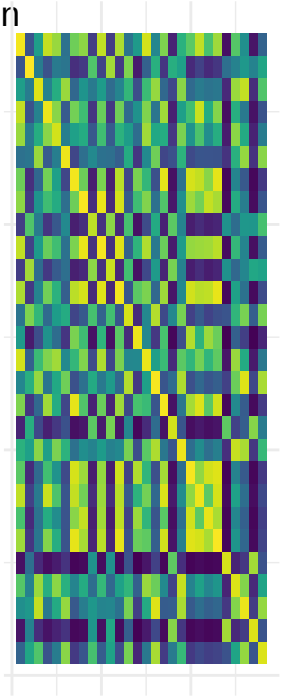


Regional similarity matrix

100 points per region

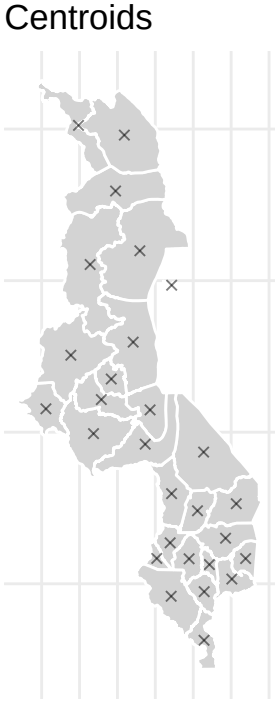


Sampled points

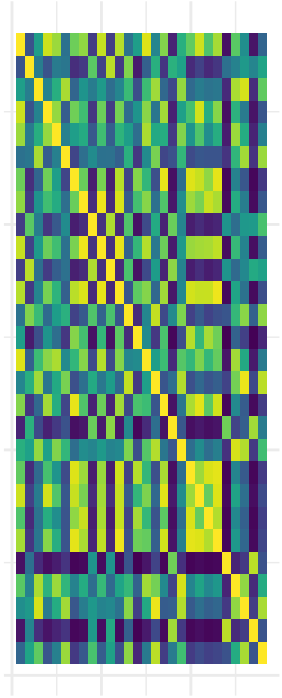


Regional similarity matrix

Centroids



Sampled points



Regional similarity matrix

```r
# Simple metric between matrices of identical dimension
matrix_metric <- function(M1, M2) {
  diff <- M1 - M2
  sum(diff^2)
}

matrix_comparison <- outer(covs, covs, Vectorize(matrix_metric))

matrix_comparison
```

```
##          [,1]      [,2]      [,3]      [,4]
## [1,] 0.000000 3.6278530 3.5640260 3.5646497
## [2,] 3.627853 0.0000000 0.3452913 0.5057681
## [3,] 3.564026 0.3452913 0.0000000 0.1371970
## [4,] 3.564650 0.5057681 0.1371970 0.0000000
```