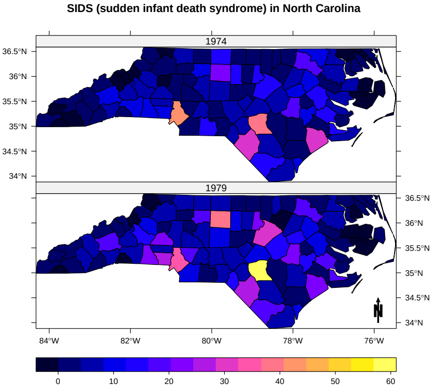
Spatial Statistics: Areal Data

Samuel I. Berchuck
STA 440L, Duke University

March 2, 2023

Goal of an Areal Data Spatial Analysis

- Account for noise due to spatial variability in order to provide smoothed estimates across space



Spatial Correlation: Areal Data

- How to induce spatial correlation between areal units?

Spatial Correlation: Areal Data

- How to induce spatial correlation between areal units?
  - Distances between centroids (possibly population weighted); may be inappropriate for oddly shaped regions of varying sizes (great for equal sized grid though)

Spatial Correlation: Areal Data

- How to induce spatial correlation between areal units?
  - Distances between centroids (possibly population weighted); may be inappropriate for oddly shaped regions of varying sizes (great for equal sized grid though)
  - Neighborhood structure of your spatial region; are two regions neighbors?

Spatial Correlation: Areal Data

- How to induce spatial correlation between areal units?
  - Distances between centroids (possibly population weighted); may be inappropriate for oddly shaped regions of varying sizes (great for equal sized grid though)
  - Neighborhood structure of your spatial region; are two regions neighbors?
- Correlation introduced through spatial random effects

Bayesian Model Fitting

- CAR models are preferred in the Bayesian setting due to the conditional definition of the model

Bayesian Model Fitting

- CAR models are preferred in the Bayesian setting due to the conditional definition of the model
- θ_i parameters are typically updated individually

Bayesian Model Fitting

- CAR models are preferred in the Bayesian setting due to the conditional definition of the model
- θ_i parameters are typically updated individually
- Depending on the likelihood, conjugacy may be available, useful for Gibbs sampling

Bayesian Model Fitting

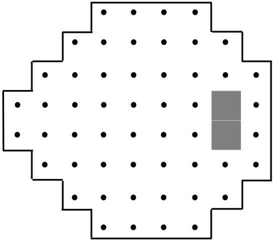
- CAR models are preferred in the Bayesian setting due to the conditional definition of the model
- θ_i parameters are typically updated individually
- Depending on the likelihood, conjugacy may be available, useful for Gibbs sampling
- Today, CAR models can be implemented using probabilistic programming languages (e.g., Stan)

Spatial Statistics

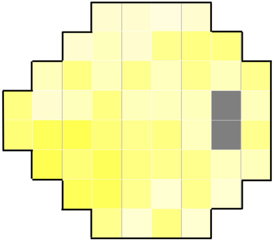
- The study of spatially referenced data observations
- Types of spatial data
  1. Geostatistical (or point-referenced)
  2. Areal (or lattice)
  3. Point-process

- The foundational assumption in spatial statistics states that dependence between observations weakens as the distance between locations increases

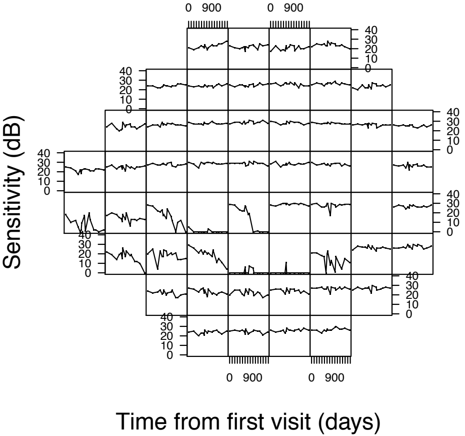
Geostatistical



Areal



Motivating an Areal Data Model



Regression Model at Each Location

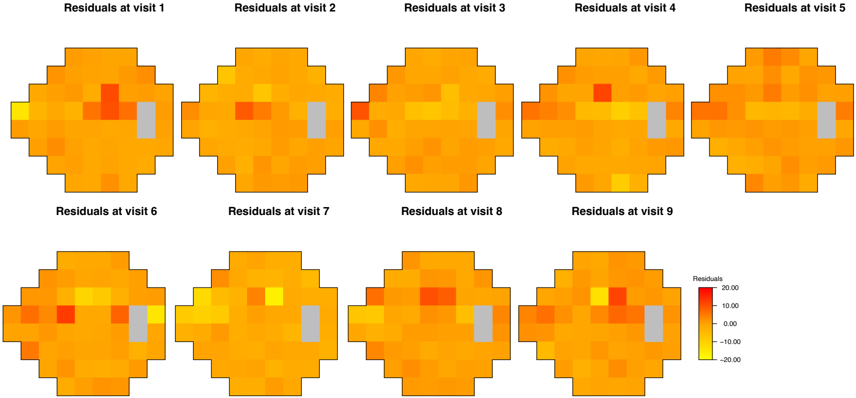
- Observation: Y_{it} is an observation at time t ($t = 1, \ldots, T$) and location $\mathbf{s}_i$ ($i = 1, \ldots, n$)
- $\mathbf{s}_i = (x_i, y_i)$ indicates the centroid of each location
- Time: x_{it} indicates the number of days from the baseline visit (i.e., $x_{i1} = 0$)
- Independent linear regression model for each location as follows,

$$Y_{it} = \tilde{\beta}_{0i} + \tilde{\beta}_{1i} x_{it} + \epsilon_{it}, \quad \epsilon_{it} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_i^2) \quad \text{for } t = 1, \ldots, T$$

- Collection of Parameters: $\boldsymbol{\Omega}_i = (\tilde{\beta}_{0i}, \tilde{\beta}_{1i}, \sigma_i^2)$
- Let's Assume a Bayesian Setup: $f(\boldsymbol{\Omega}_i) = f(\tilde{\beta}_{0i}) f(\tilde{\beta}_{1i}) f(\sigma_i^2)$
- Prediction: $f(Y_{it'} | \mathbf{Y}_i) = \int f(Y_{it'} | \boldsymbol{\Omega}_i) f(\boldsymbol{\Omega}_i | \mathbf{Y}_i) d\boldsymbol{\Omega}_i$

What are some potential problems with this setup?

Residuals from PLR Setup



Joint Inference Across Locations

- Define a joint model for $t = 1, \ldots, T$ and $i = 1, \ldots, n$:

$$
\begin{aligned}
Y_{it} &= \tilde{\beta}_{0i} + \tilde{\beta}_{1i} x_{it} + \epsilon_{it}, \quad \epsilon_{it} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_i^2) \\
&= (\beta_0 + \beta_{0i}) + (\beta_1 + \beta_{1i}) x_{it} + \epsilon_{it} \\
&= (\beta_0 + \beta_1 x_{it}) + (\beta_{0i} + \beta_{1i} x_{it}) + \epsilon_{it} \\
&= \mathbf{x}_{it} \boldsymbol{\beta} + \mathbf{x}_{it} \boldsymbol{\beta}_i + \epsilon_{it},
\end{aligned}
$$

where $\mathbf{x}_{it} = (1, x_{it})$, $\boldsymbol{\beta} = (\beta_0, \beta_1)^\top$, and $\boldsymbol{\beta}_i = (\beta_{0i}, \beta_{1i})^\top$.

- $\boldsymbol{\beta}$: population regression parameter
- $\boldsymbol{\beta}_i$: location-specific parameters
- σ_i^2: location-specific variance

Let's treat the intercept and slopes independently:

- $\boldsymbol{\beta}_0 = (\beta_{01}, \ldots, \beta_{0n})^\top$, $\boldsymbol{\beta}_1 = (\beta_{11}, \ldots, \beta_{1n})^\top$

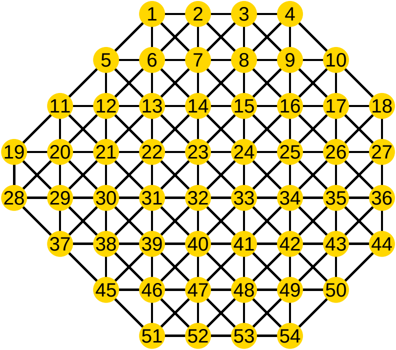
Our goal is to apply the fundamental assumption in spatial statistics to learning the location-specific parameters.

Neighborhood Adjacency Matrix

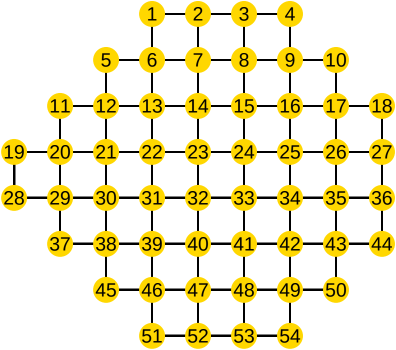
- Spatial dependency is encoded through an $n \times n$ adjacency matrix, $\mathbf{W}$.
- Each entry is given by:
  $$w_{ij} = 1(i \sim j) = \begin{cases} 1 & \text{if } i \text{ and } j \text{ share a border;} \\ 0 & \text{otherwise.} \end{cases}$$
- Border definition: queen (edges and corners) or rook (edges only) specification

```
library(womblR)
W1 <- womblR::HFAII_Queen # HFA-II visual field adjacency matrix queen
W2 <- womblR::HFAII_Rook # HFA-II visual field adjacency matrix rook
W3 <- womblR::HFAII_QueenHF # HFA-II visual field adjacency matrix queen with hemisphere
```

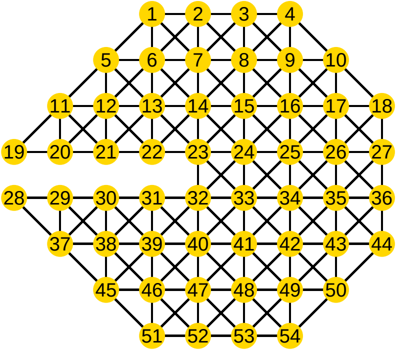
Queen Specification



Rook Specification



Queen Specification with Hemisphere Separation



Inducing Spatial Dependency

The most common approach for inducing spatial dependency in an areal data setting using the intrinsic conditional autoregressive (ICAR) process,

$$\boldsymbol{\theta}|\tau^2 \sim \text{ICAR}\left(\tau^2\right)$$

$$\theta_i|\boldsymbol{\theta}_{-i}, \tau^2 \sim \mathcal{N}\left(\frac{\sum_{j=1}^n w_{ij}\theta_j}{\sum_{j=1}^n w_{ij}}, \frac{\tau^2}{\sum_{j=1}^n w_{ij}}\right)$$

- $\boldsymbol{\theta}_{-j}$: Vector of θ_i parameters with θ_j removed
- $\boldsymbol{\beta}_0|\tau_0^2 \sim \text{ICAR}\left(\tau_0^2\right), \quad \boldsymbol{\beta}_1|\tau_1^2 \sim \text{ICAR}\left(\tau_1^2\right)$
- Not a proper prior distribution (implications for MCMC algorithms!)

A Proper CAR Process

$$\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_n\}^{\mathsf{T}}\,;\ \boldsymbol{\theta}|\rho, \tau^2 \sim \text{Leroux CAR}\left(\rho, \tau^2\right)$$

$$\theta_i|\boldsymbol{\theta}_{-i}, \rho, \tau^2 \sim \mathcal{N}\left(\frac{\rho \sum_{j=1}^{n} w_{ij}\theta_j}{\rho \sum_{j=1}^{n} w_{ij} + 1 - \rho}, \frac{\tau^2}{\rho \sum_{j=1}^{n} w_{ij} + 1 - \rho}\right)$$

- Proper for $\rho \in [0, 1)$
- $\rho = 1$ gives us the ICAR model
- ρ is given a prior distribution and estimated by the data
- $\text{COR}(\theta_k, \theta_j|\boldsymbol{\theta}_{-kj}) = \frac{\rho w_{kj}}{\sqrt{(\rho \sum_{i=1}^{n} w_{ki} + 1 - \rho)(\rho \sum_{i=1}^{n} w_{ji} + 1 - \rho)}}$

Equivalency of Joint and Conditional Specifications

$$\theta_i|\boldsymbol{\theta}_{-i}, \rho, \tau^2 \sim \mathcal{N}\left(\frac{\rho \sum_{j=1}^n w_{ij}\theta_j}{\rho \sum_{j=1}^n w_{ij} + 1 - \rho}, \frac{\tau^2}{\rho \sum_{j=1}^n w_{ij} + 1 - \rho}\right)$$

$$\iff$$

$$\boldsymbol{\theta}|\rho, \tau^2 \sim \mathcal{N}(\mathbf{0}, \tau^2\mathbf{Q}(\rho)^{-1}), \quad \mathbf{Q}(\rho) = \rho\left(\mathbf{D}_w - \mathbf{W}\right) + (1-\rho)\mathbf{I}_n,$$

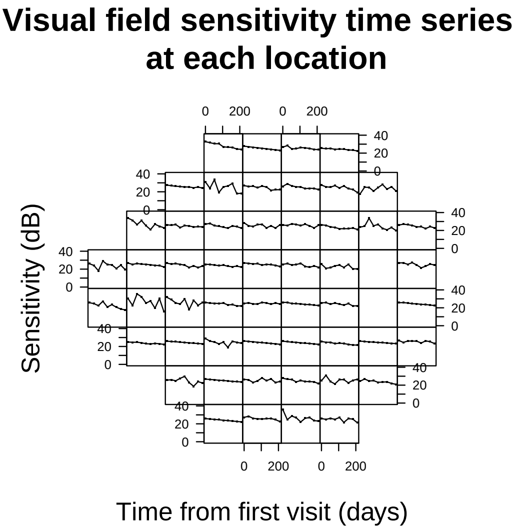
where $\mathbf{D}_w$ is a diagonal matrix with the rowsum of $\mathbf{W}$ on the diagonal.

Let's fit this model using stan!

First let's simulate some data:

```
###Simulation settings
set.seed(2)
T <- 15 # number of time points
n <- 54 # number of spatial locations
N <- n * T # total number of observations
W <- HFAII_QueenHF # visual field adjacency matrix

###Generating model
D <- diag(apply(W, 1, sum))
Q <- 0.99 * (D - W) + 0.01 * diag(n)
precision0 <- 1.4 * Q
sigma0 <- solve(precision0)
beta0 <- 26 + chol(sigma0) %*% matrix(rnorm(n), ncol = 1)
precision1 <- 0.5 * Q
sigma1 <- solve(precision1)
beta1 <- -6 + chol(sigma1) %*% matrix(rnorm(n), ncol = 1)
sigma <- rhalft(n, scale = 1.5, nu = 3) # requires the LaplacesDemon package

###Simulate data
y <- matrix(nrow = n, ncol = T)
time <- seq(0, 1, length.out = T)
for (i in 1:n) {
  for (t in 1:T) {
    mu <- beta0[i] + beta1[i] * time[t]
    y[i, t] <- rnorm(1, mu, sigma[i])
  }
}
```

Data Visualization



Stan Model Specification

```
data {
  int<lower = 1> n; // number of spatial locations
  int<lower = 1> T; // number of longitudinal observations
  int<lower = 1> N; // total number of observations
  vector[N] y; // outcome observations
  vector[N] x; // time measurements
  int<lower = 1> s[N]; // location indeces
  matrix<lower = 0, upper = 1>[n, n] W; // adjacency matrix
}

transformed data {
  vector[n] zeros;
  matrix[n, n] identity;
  matrix<lower = 0>[n, n] D;
  vector[n] W_rowsums;
  for (i in 1:n) {
    W_rowsums[i] = sum(W[i, ]);
  }
  D = diag_matrix(W_rowsums);
  zeros = rep_vector(0, n);
  identity = diag_matrix(rep_vector(1.0, n));
}

parameters {
  real beta0;
  real beta1;
  vector[n] beta0_vec;
  vector[n] beta1_vec;
  vector<lower = 0>[n] sigma;
  real<lower = 0> tau0;
  real<lower = 0, upper = 1> rho0;
  real<lower = 0> tau1;
  real<lower = 0, upper = 1> rho1;
}
```

Stan Model Specification (Cont.)

```
transformed parameters {
  cov_matrix[n] precision0 = (1 / (tau0 * tau0)) * (rho0 * (D - W) + (1 - rho0) * identity);
  cov_matrix[n] precision1 = (1 / (tau1 * tau1)) * (rho1 * (D - W) + (1 - rho1) * identity);
}

model {
  vector[N] mu;
  for (i in 1:N) {
    mu[i] = (beta0 + beta0_vec[s[i]]) + (beta1 + beta1_vec[s[i]]) * x[i];
  }
  beta0_vec ~ multi_normal_prec(zeros, precision0);
  beta1_vec ~ multi_normal_prec(zeros, precision1);
  for (i in 1:n) {
    sigma[i] ~ student_t(3, 0, 1);
  }
  tau0 ~ student_t(3, 0, 1);
  tau1 ~ student_t(3, 0, 1);
  for (i in 1:N) {
    y[i] ~ normal(mu[i], sigma[s[i]]);
  }
}

generated quantities {
  vector[N] log_lik;
  vector[N] mu;
  for (i in 1:N) {
    mu[i] = (beta0 + beta0_vec[s[i]]) + (beta1 + beta1_vec[s[i]]) * x[i];
  }
  for (i in 1:N) log_lik[i] = normal_lpdf(y[i] | mu[i], sigma[s[i]]);
}
```

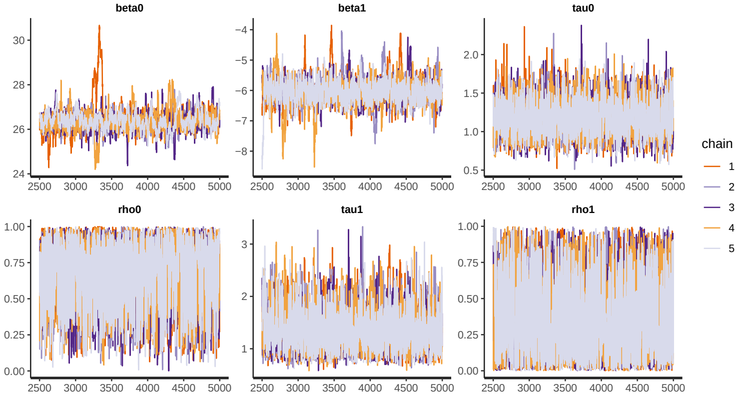
Prepare Date for stan

```
###Create training data
T_train <- 9
dat <- data.frame(y = as.numeric(y[, 1:T_train]),
   location = rep(1:n, T_train),
   time = rep(time[1:T_train], each = n))

###Define stan data list
blind_spot <- c(26, 35) # define blind spot
dat <- dat[!dat$location %in% blind_spot, ]
W <- W[-blind_spot, -blind_spot]
n_train <- 52
dat$location <- as.numeric(as.factor(dat$location)) # reorder the locations (1-52)
stan_data <- list(
  y = dat$y,
  x = dat$time,
  s = dat$location,
  N = T_train * n_train,
  n = n_train,
  T = T_train,
  W = W
)
```

We will only fit the model with the first nine observations. We will use the remaining six to look at prediction performance.

Compile model and perform sampling

```
###Compile spatial model
model_compiled <- stan_model("spatial.stan")
fit <- sampling(model_compiled, data = stan_data, chains = 5, iter = 5000, cores = 5)
saveRDS(fit, file = "fit.rds")

###Check the traceplots
rstan::traceplot(fit, pars = c("beta0", "beta1", "tau0", "rho0", "tau1", "rho1"))
```



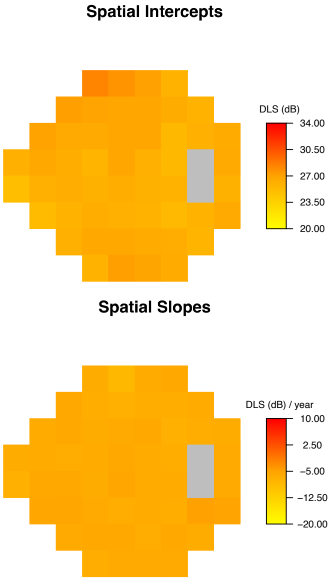
Check model fit

We can check model fit using the WAIC (also known as the Wattanabe-Akaike information criterion, or the widely applicable information criterion).

```
###First let's fit a non-spatial version of our model (the file "plr.stan" is available on Sakai)
model_compiled_plr <- stan_model("plr.stan")
fit_plr <- sampling(model_compiled_plr, data = stan_data, chains = 5, iter = 5000, cores = 5)
saveRDS(fit_plr, file = "fit_plr.rds")

###Compute waic for spatial model
library(loo)
log_lik <- loo::extract_log_lik(fit)
waic <- loo::waic(log_lik)
waic$estimates[3, 1]
[1] 1300.374

###Compute waic for plr model
log_lik_plr <- loo::extract_log_lik(fit_plr)
waic_plr <- loo::waic(log_lik_plr)
waic_plr$estimates[3, 1]
[1] 1324.804
```

The WAIC for the spatial model is lower, indicating a better fit

Posterior Intercepts and Slopes at Each Location



Prediction performance: average MSE across locations

We use the posterior predictive distribution:

$$f\left(Y_{it}|\mathbf{Y}\right) = \int f\left(Y_{it}, \mathbf{\Omega}|\mathbf{Y}\right) d\mathbf{\Omega} = \int f\left(Y_{it}|\mathbf{\Omega}\right) f\left(\mathbf{\Omega}|\mathbf{Y}\right) d\mathbf{\Omega}$$