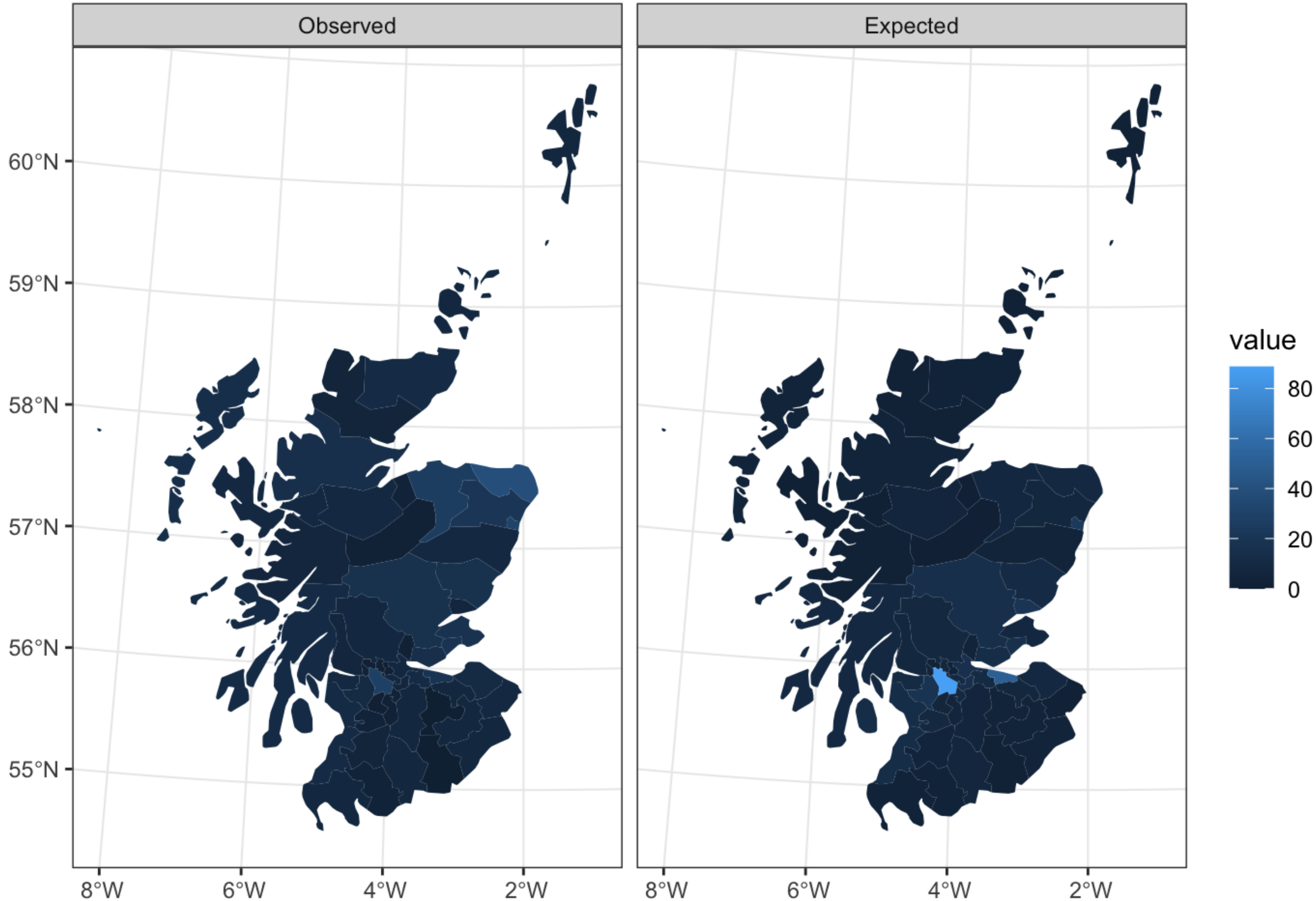
Spatial GLM + Point Reference Spatial Data

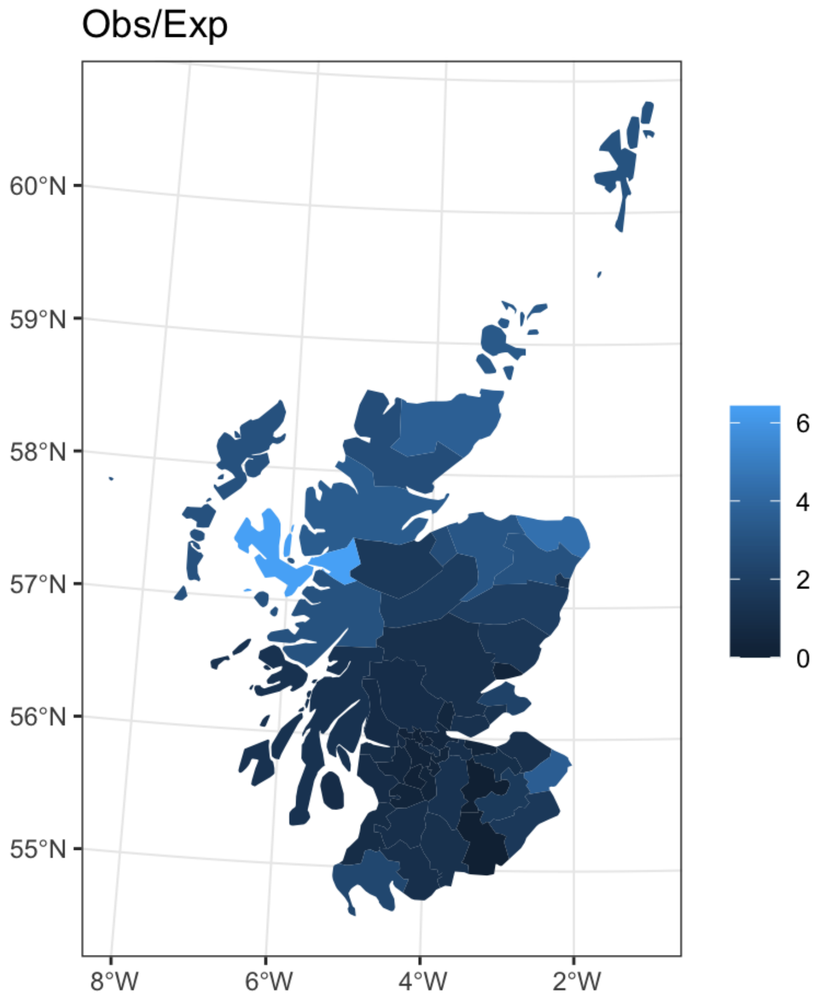
Lecture 21

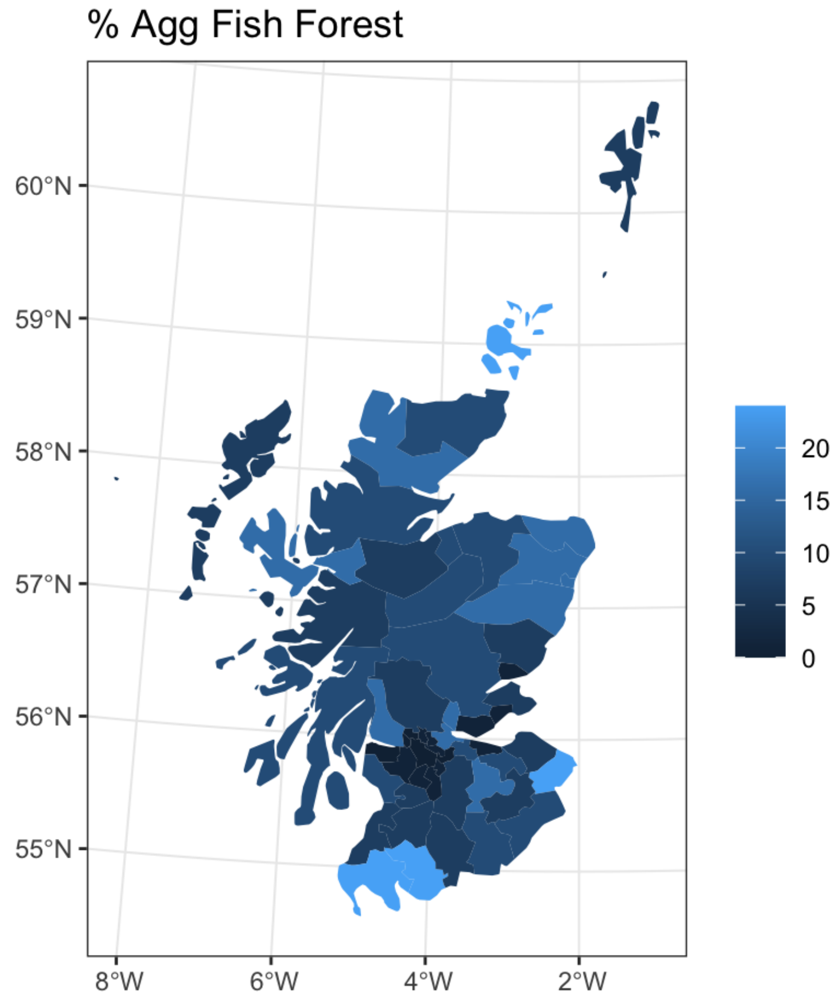
Dr. Colin Rundel

Spatial GLM Models

Scottish Lip Cancer Data

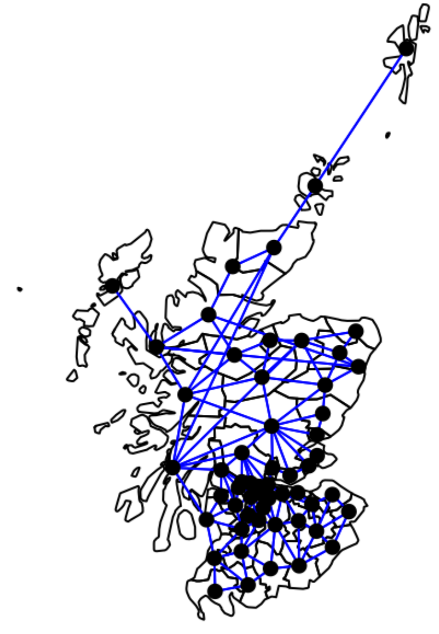



Obs/Exp
60°N
59°N
58°N
57°N
56°N
55°N
8°W
6°W
4°W
2°W
6
4
2
0


% Agg Fish Forest
60°N
59°N
58°N
57°N
56°N
55°N
8°W
6°W
4°W
2°W
20
15
10
5
0

Neighborhood / weight matrix

```
A = (st_distance(lip_cancer) |> unclass()) < 1e-6
listw = spdep::mat2listw(A)
```



Moran's I

```r
spdep::moran.test(lip_cancer$Observed, listw)
```

```
	Moran I test under randomisation

data:  lip_cancer$Observed  
weights: listw    

Moran I statistic standard deviate = 4.5416,
p-value = 2.792e-06
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation 
      0.311975396      -0.018181818 
         Variance 
      0.005284831 
```

```r
spdep::moran.test(lip_cancer$Observed / lip_c
```

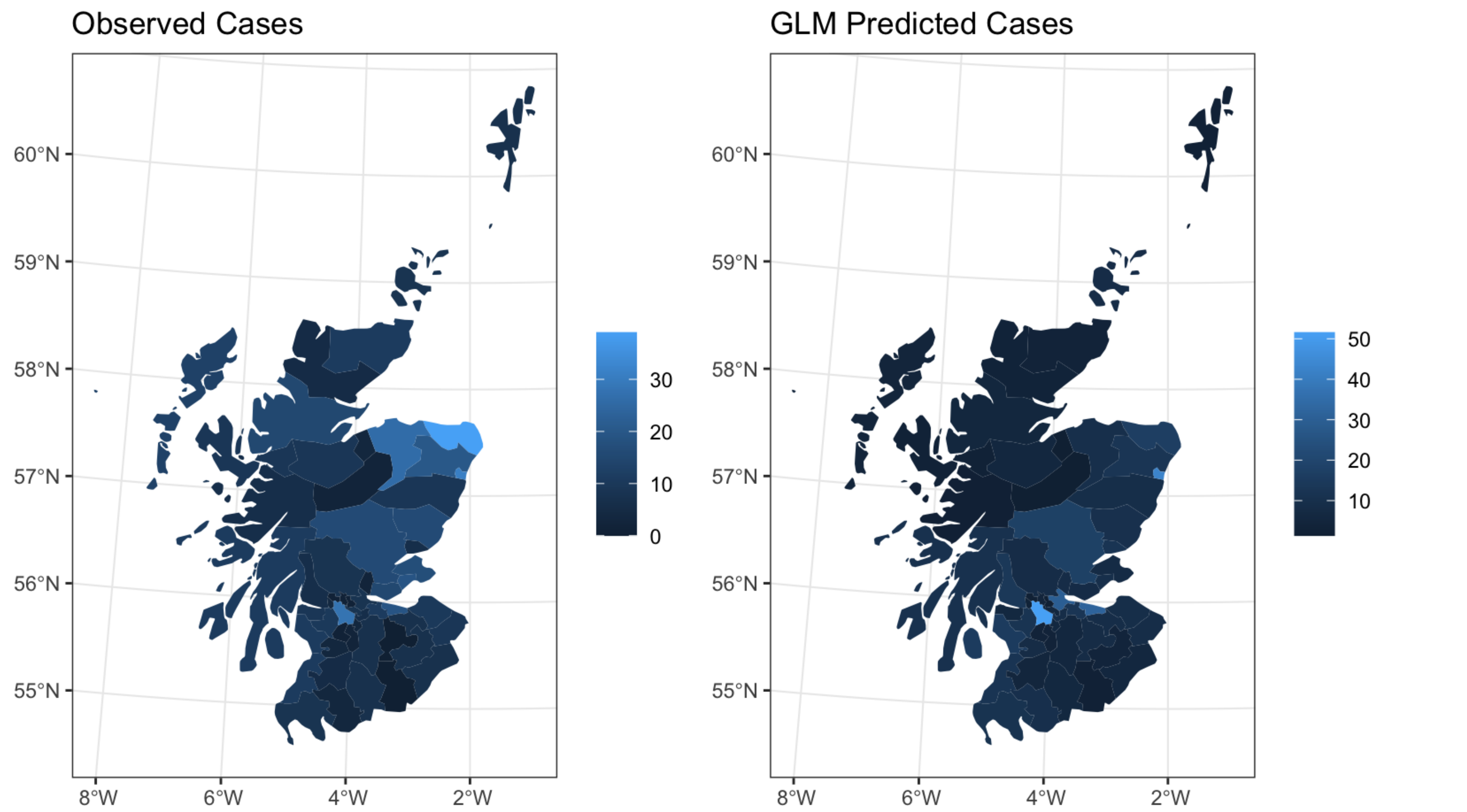
```
	Moran I test under randomisation

data:  lip_cancer$Observed/lip_cancer$Expected  
weights: listw    

Moran I statistic standard deviate = 8.2916,
p-value < 2.2e-16
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation 
      0.589795225      -0.018181818 
         Variance 
      0.005376506 
```

GLM

```r
l = glm(Observed ~ offset(log(Expected)) + pcaff,
        family="poisson", data=lip_cancer)
summary(l)
```

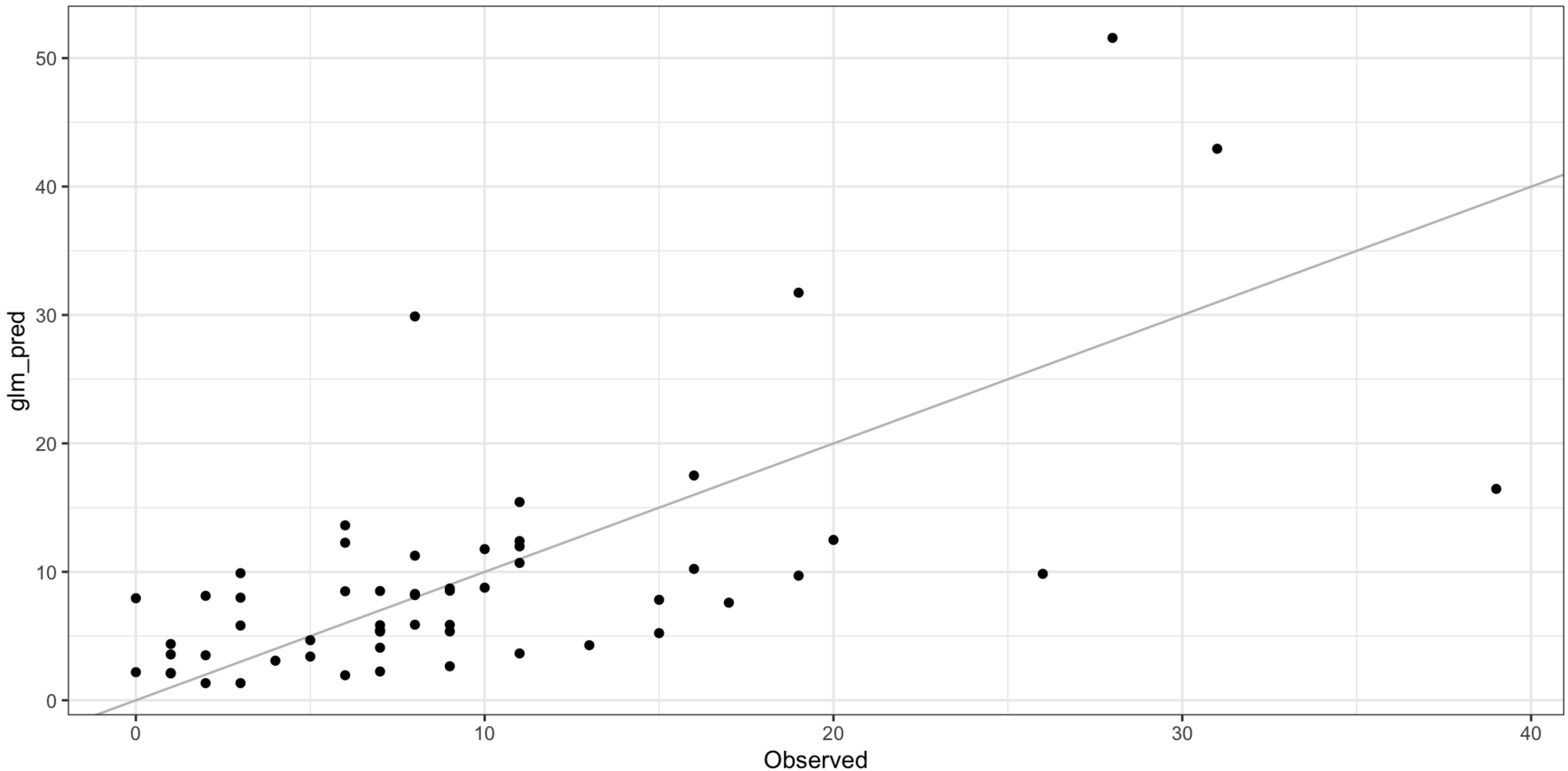
```

Call:
glm(formula = Observed ~ offset(log(Expected)) + pcaff, family = "poisson",
    data = lip_cancer)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.7632  -1.2156   0.0967   1.3362   4.7130

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.542268   0.069525   -7.80 6.21e-15
pcaff        0.073732   0.005956   12.38  < 2e-16

(Intercept) ***
pcaff       ***
```

GLM Fit

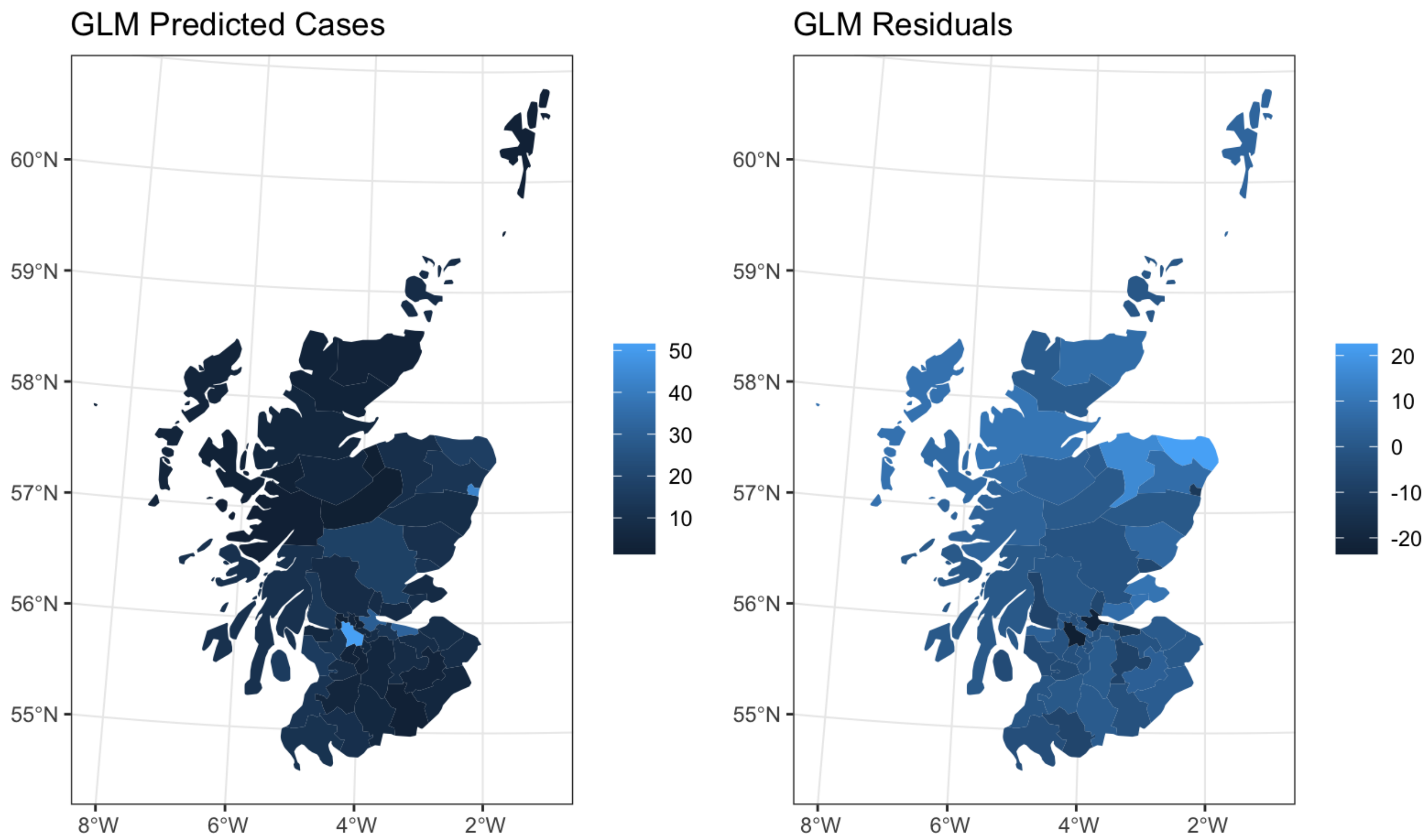


Observed Cases
GLM Predicted Cases
60°N
59°N
58°N
57°N
56°N
55°N
8°W
6°W
4°W
2°W
0
10
20
30
40
50

GLM Fit



GLM Residuals



Model Results

```r
#RMSE
yardstick::rmse_vec(lip_cancer$Observed, lip_cancer$glm_pred)
```

```
[1] 7.480889
```

```r
#Moran's I
spdep::moran.test(lip_cancer$glm_resid, listw)
```

```

	Moran I test under randomisation

data:  lip_cancer$glm_resid  
weights: listw    

Moran I statistic standard deviate = 4.8186,
p-value = 7.228e-07
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation 
      0.333403223      -0.018181818 
         Variance 
      0.005323717 
```

A hierachical model for lip cancer

We have observed counts of lip cancer for 56 districts in Scotland. Let y_i represent the number of lip cancer for district i.

$$y_i \sim \text{Poisson}(\lambda_i)$$

$$\log(\lambda_i) = \log(E_i) + x_i\beta + \omega_i$$

$$\boldsymbol{\omega} \sim \quad (\mathbf{0},\ \sigma^2(\boldsymbol{D} - \phi\,\boldsymbol{A})^{-1})$$

where E_i is the expected counts for each region (and serves as an offset).

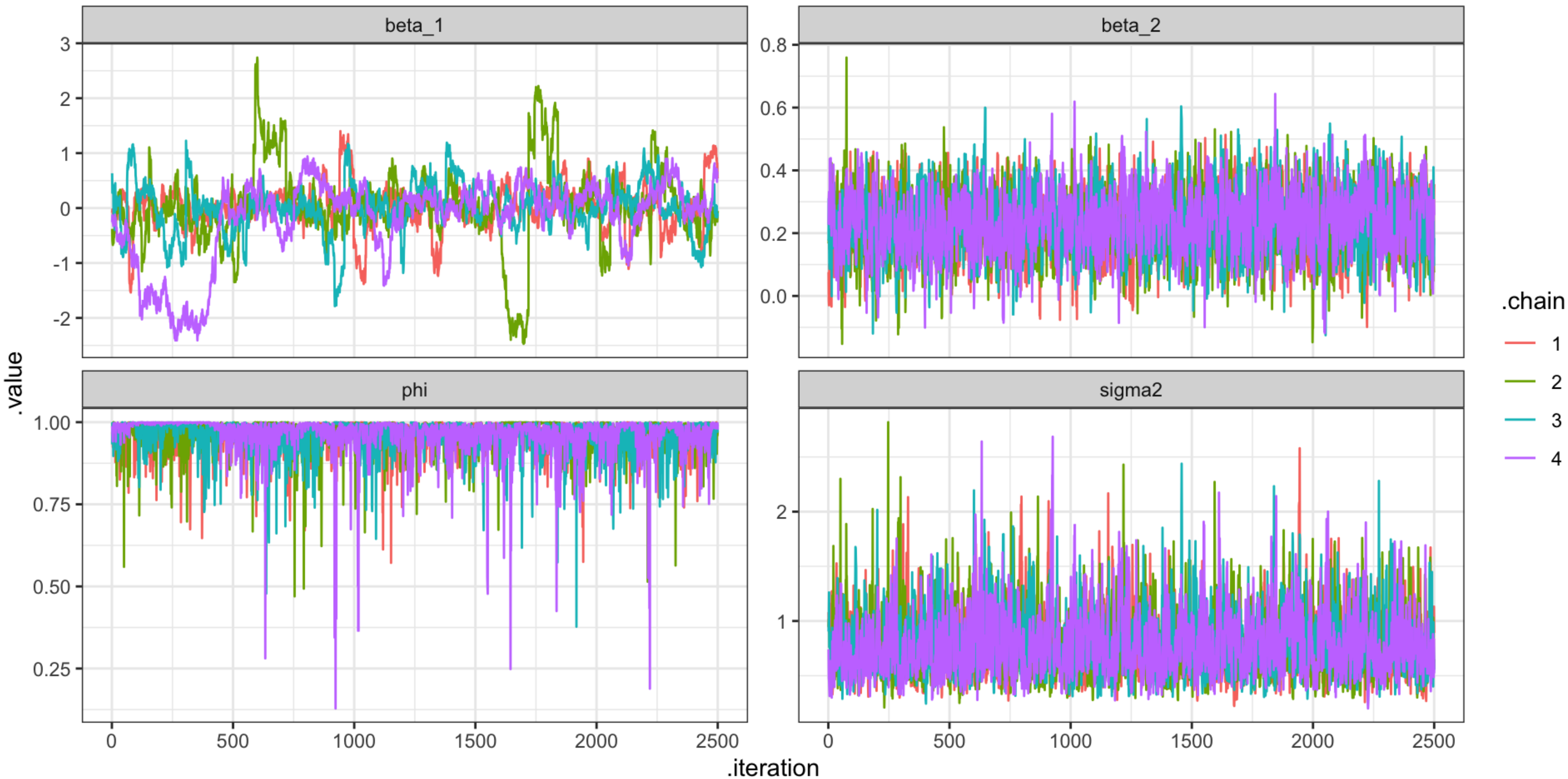
Data prep & CAR model

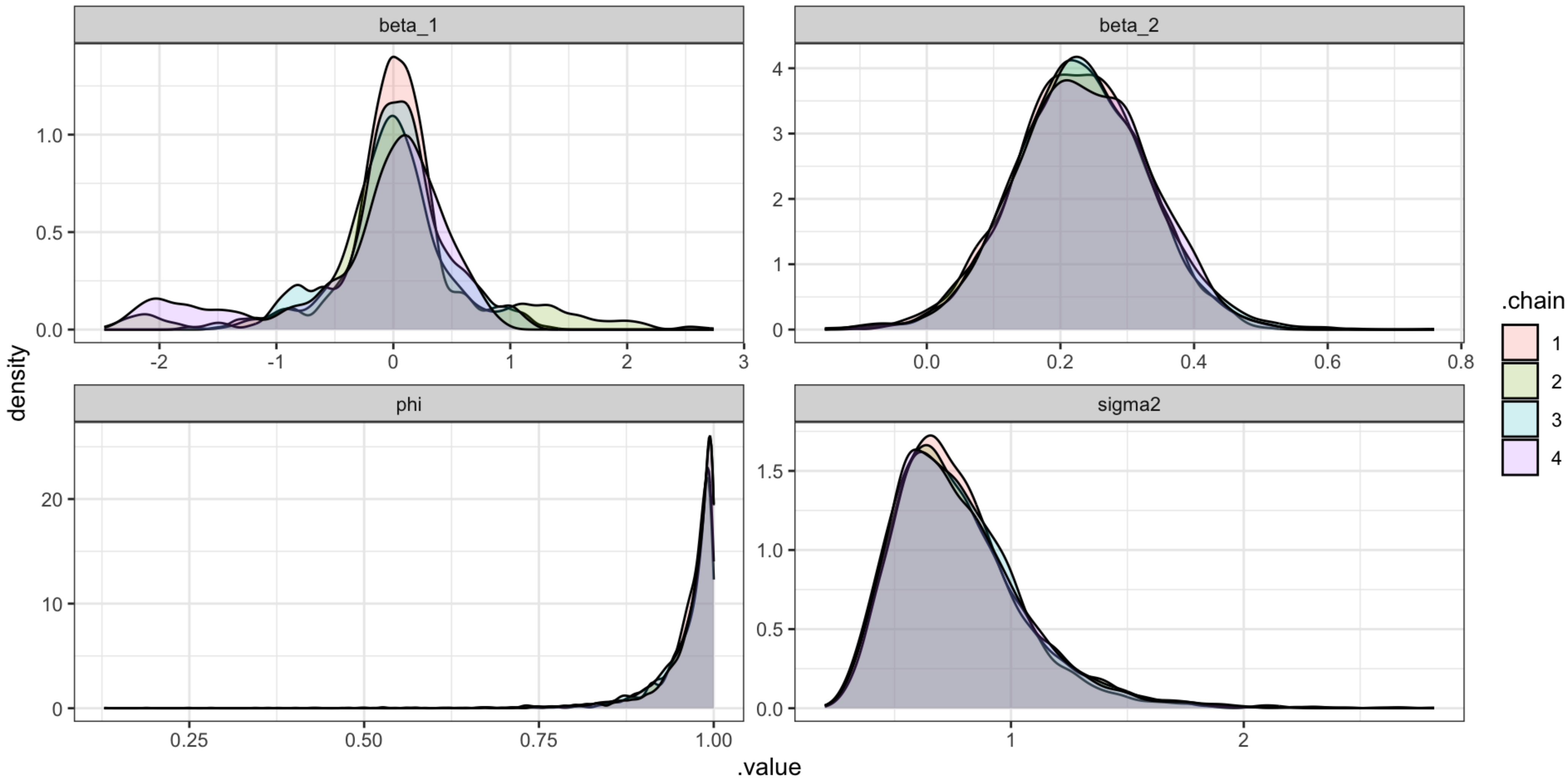
```
X = model.matrix(~scale(lip_cancer$pcaff))
offset = lip_cancer$Expected
y = lip_cancer$Observed
```

```
car_model = "
data {
  int<lower=0> N;
  int<lower=0> p;
  int<lower=0> y[N];
  matrix[N,N] A;
  matrix[N,p] X;
  vector[N] offset;
}
transformed data {
  vector[N] nb = A * rep_vector(1, N);
  matrix[N,N] D = diag_matrix(nb);
}
parameters {
  vector[N] w_s;
  vector[p] beta;
  real<lower=0> sigma2;
  real<lower=0,upper=1> phi;
}
transformed parameters {
  vector[N] eta = log(offset) + X * beta + w_s;
```

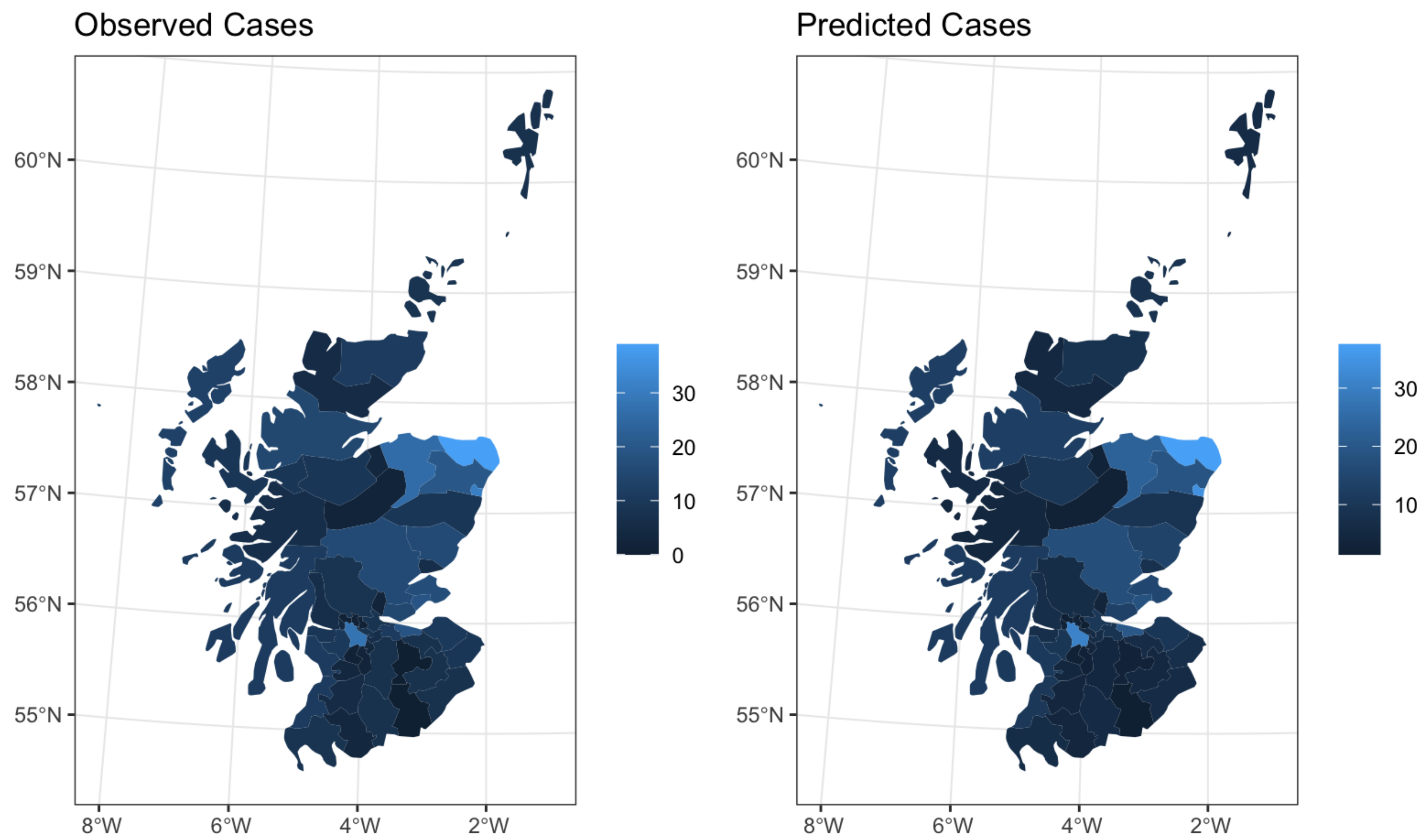
CAR Fitting

```
car = rstan::stan_model(model_code = car_model)

car_m = rstan::sampling(
  car, iter=5000, cores=4,
  data = list(N=nrow(X), A=A, X=X, p=ncol(X), offset=offset, y=y)
)
```

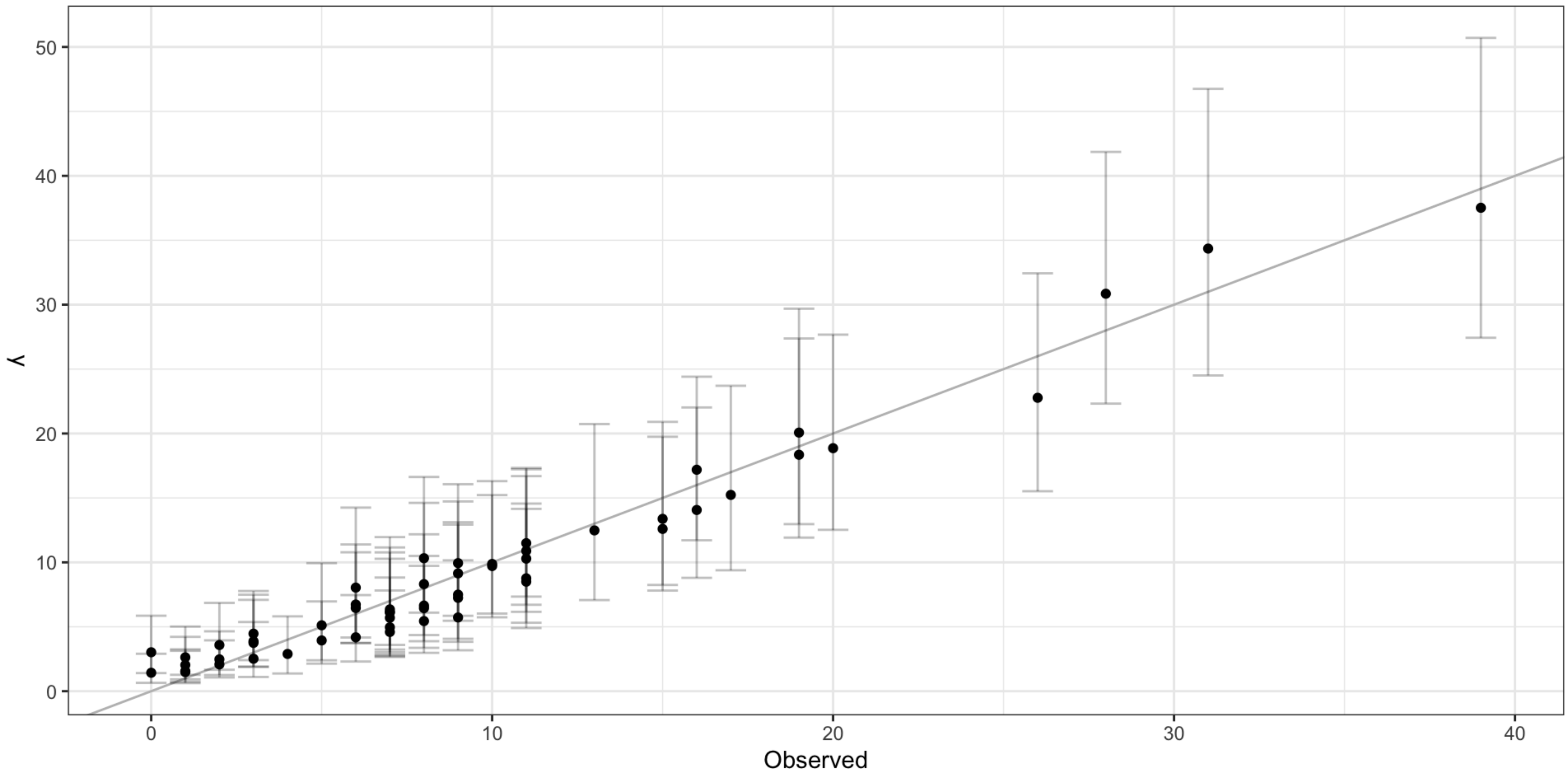

beta_1
beta_2
phi
sigma2
.value
.iteration
.chain
1
2
3
4
0
500
1000
1500
2000
2500


beta_1
beta_2
phi
sigma2
density
.value
.chain
1
2
3
4

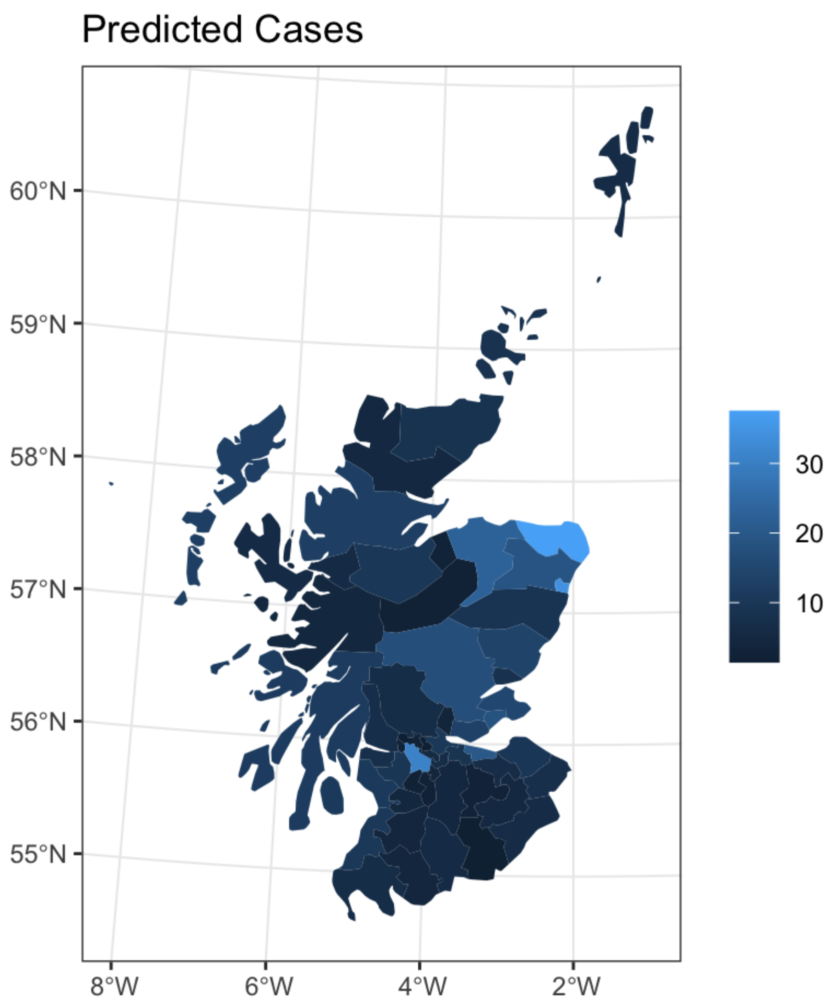
CAR Predictions ($\hat{\lambda}$)

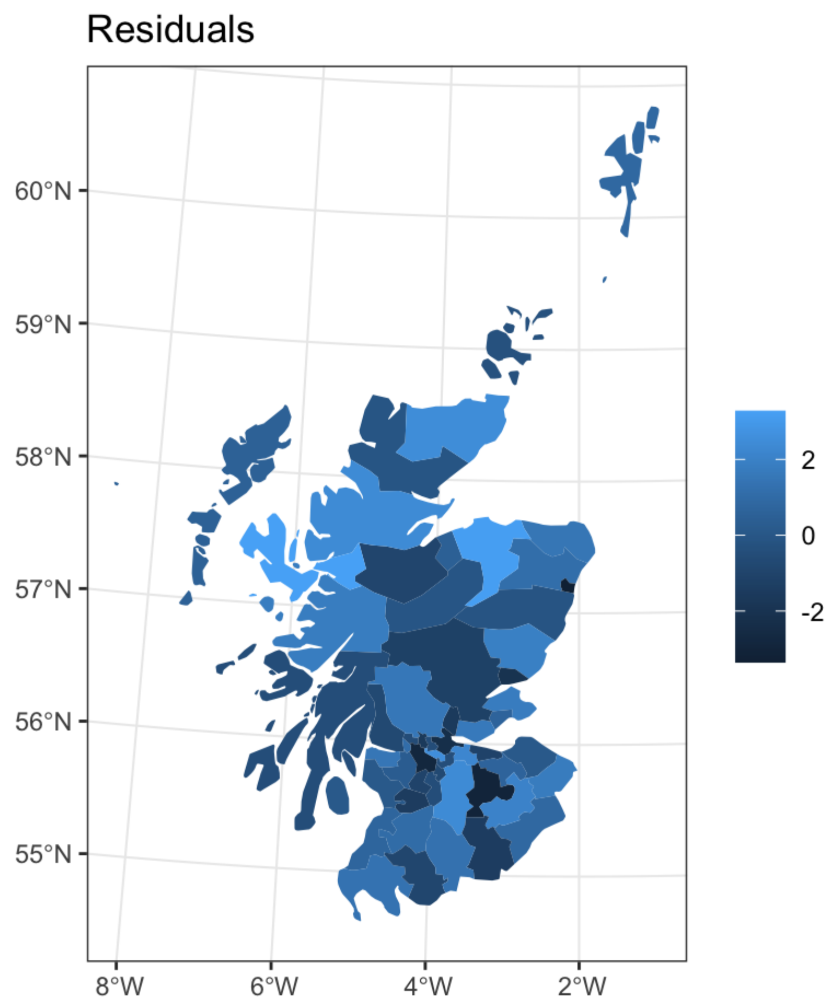


CAR Predictions



CAR Residuals





CAR Results

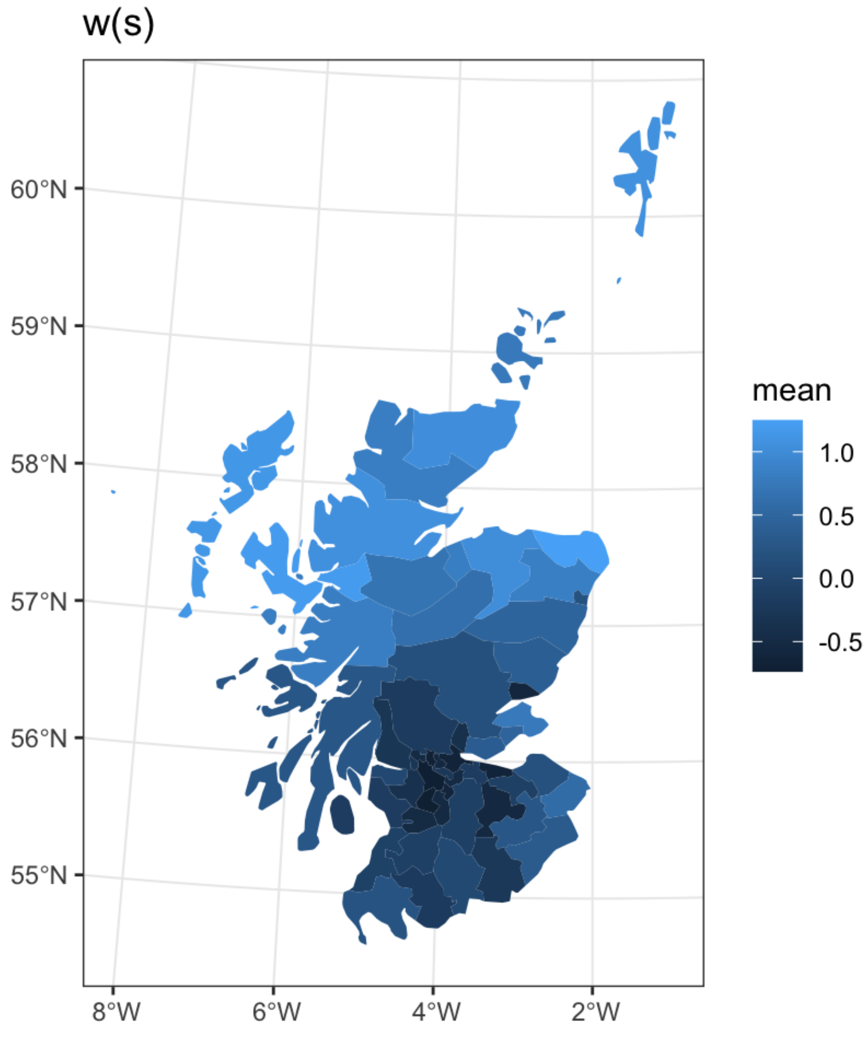
```
#RMSE
yardstick::rmse_vec(car_lip_cancer$Observed, car_lip_cancer$y_pred)
```

```
[1] 1.599187
```

```
#Moran's I
spdep::moran.test(car_lip_cancer$Observed - car_lip_cancer$y_pred, listw)
```

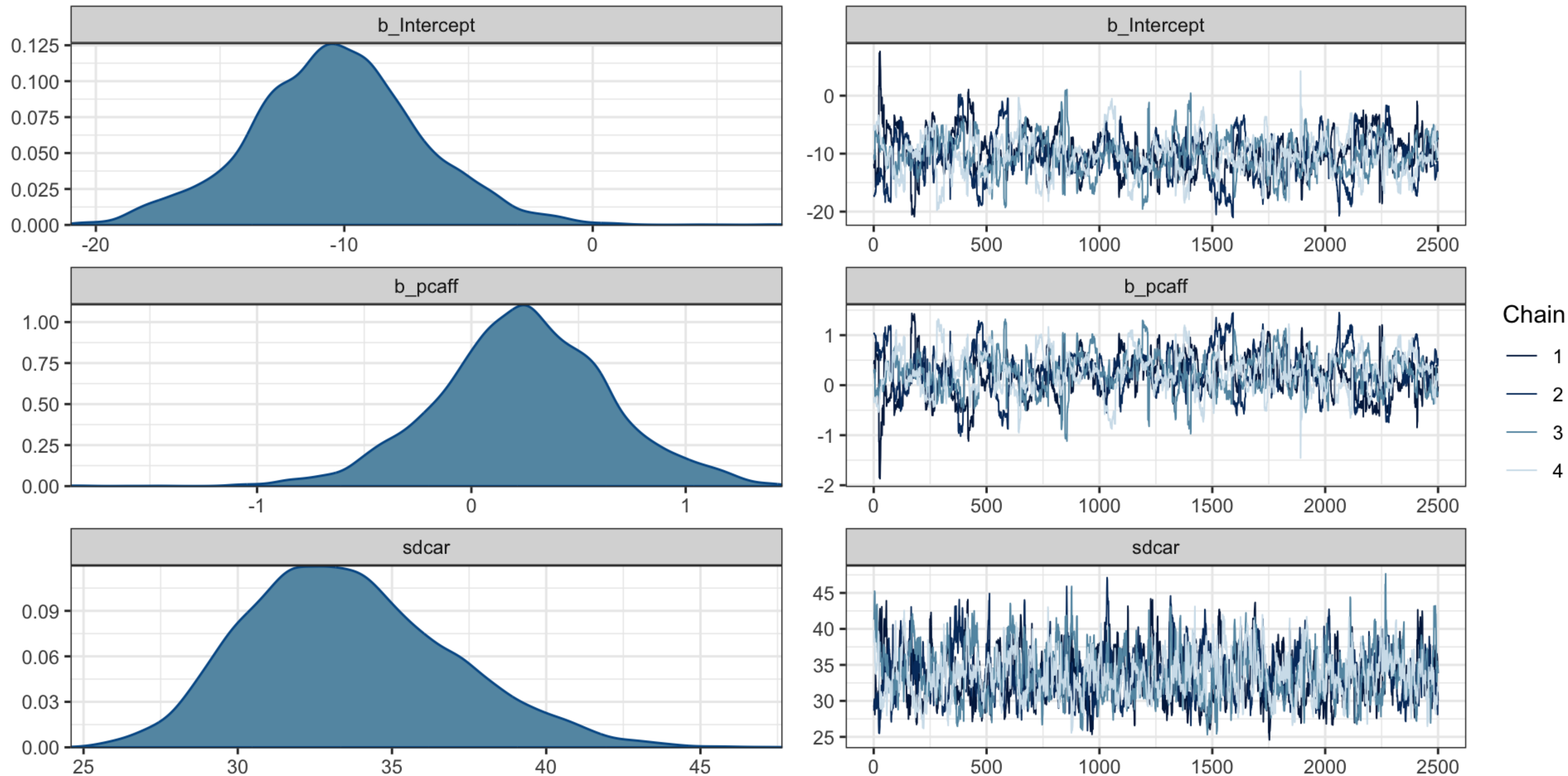
```

	Moran I test under randomisation

data:  car_lip_cancer$Observed - car_lip_cancer$y_pred
weights: listw

Moran I statistic standard deviate =
0.73353, p-value = 0.2316
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation
      0.036963635      -0.018181818
         Variance
      0.005651802
```

Latent spatial process



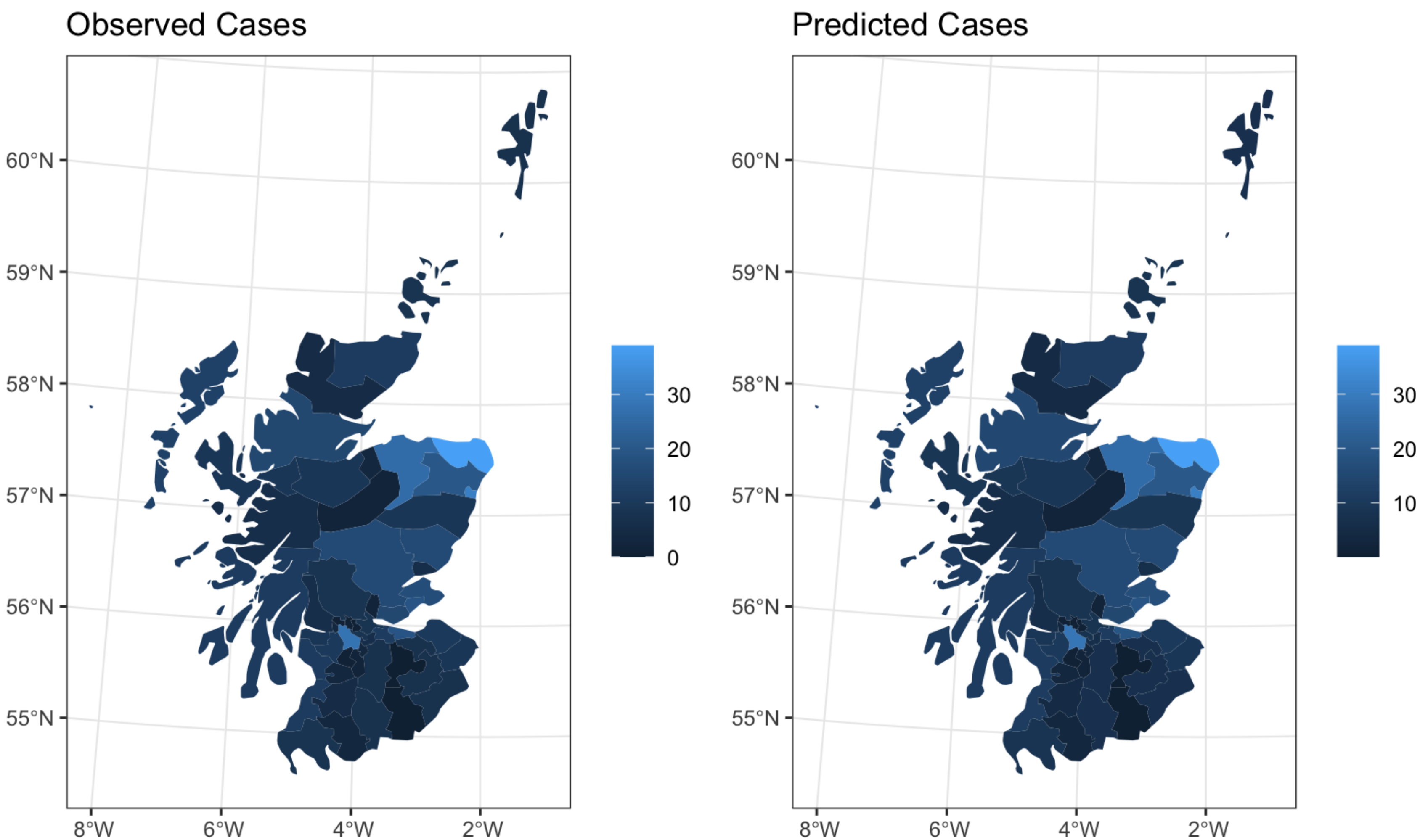
Intrinsic Autoregressive Model (IAR)

```r
rownames(A) = lip_cancer$District

iar_m = brms::brm(
  Observed~offset(Expected)+pcaff+car(A, gr=District, type="icar"),
  data=lip_cancer, data2=list(A=A),
  family = poisson, cores=4, iter=5000
)
```

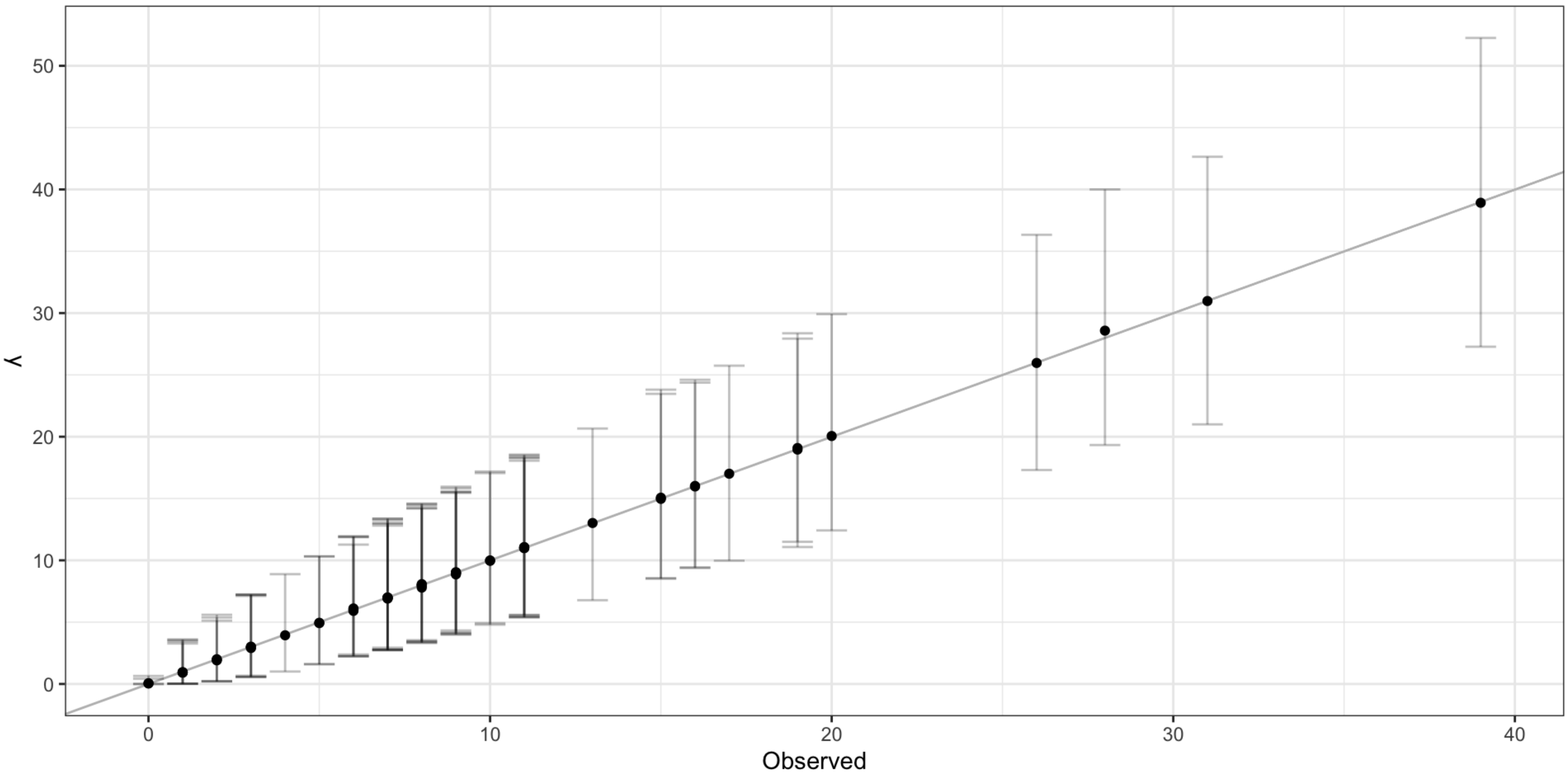


b_Intercept
0.125
0.100
0.075
0.050
0.025
0.000
-20
-10
0
b_Intercept
0
-10
-20
0
500
1000
1500
2000
2500
b_pcaff
1.00
0.75
0.50
0.25
0.00
-1
0
1
b_pcaff
1
0
-1
-2
0
500
1000
1500
2000
2500
Chain
1
2
3
4
sdcar
0.09
0.06
0.03
0.00
25
30
35
40
45
sdcar
45
40
35
30
25
0
500
1000
1500
2000
2500

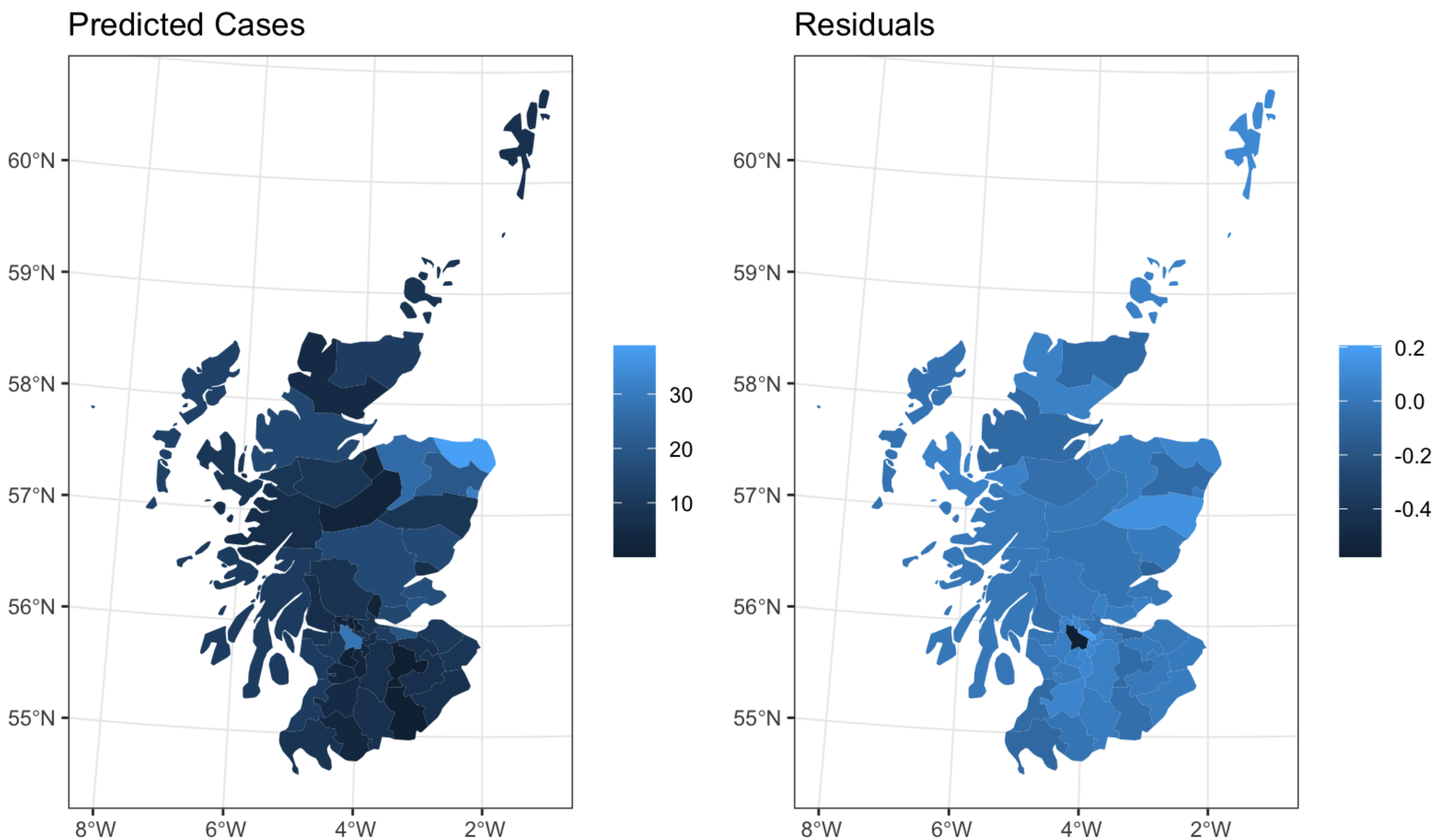
Predictions



Observed vs predicted



Residuals



Predicted Cases
Residuals
60°N
59°N
58°N
57°N
56°N
55°N
8°W
6°W
4°W
2°W
30
20
10
0.2
0.0
-0.2
-0.4

IAR Results

```
#RMSE
yardstick::rmse_vec(iar_pred$Observed, iar_pred$y_pred)
```

```
[1] 0.09762396
```

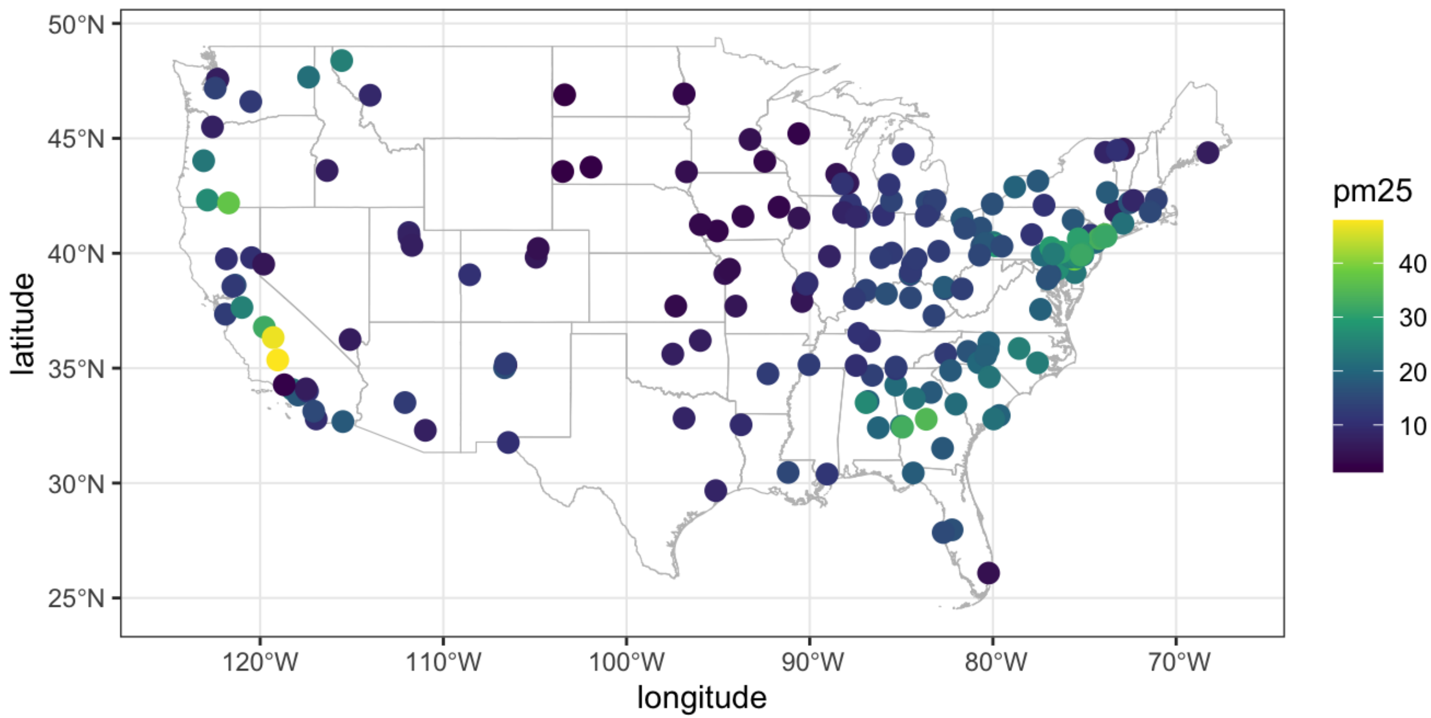
```
#Moran's I
spdep::moran.test(iar_pred$resid, listw)
```

```

    Moran I test under randomisation

data:  iar_pred$resid
weights: listw

Moran I statistic standard deviate = 2.5724,
p-value = 0.005051
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation
      0.131306700      -0.018181818
         Variance
      0.003377189
```

Point Referenced Data

Example - PM2.5 from CSN

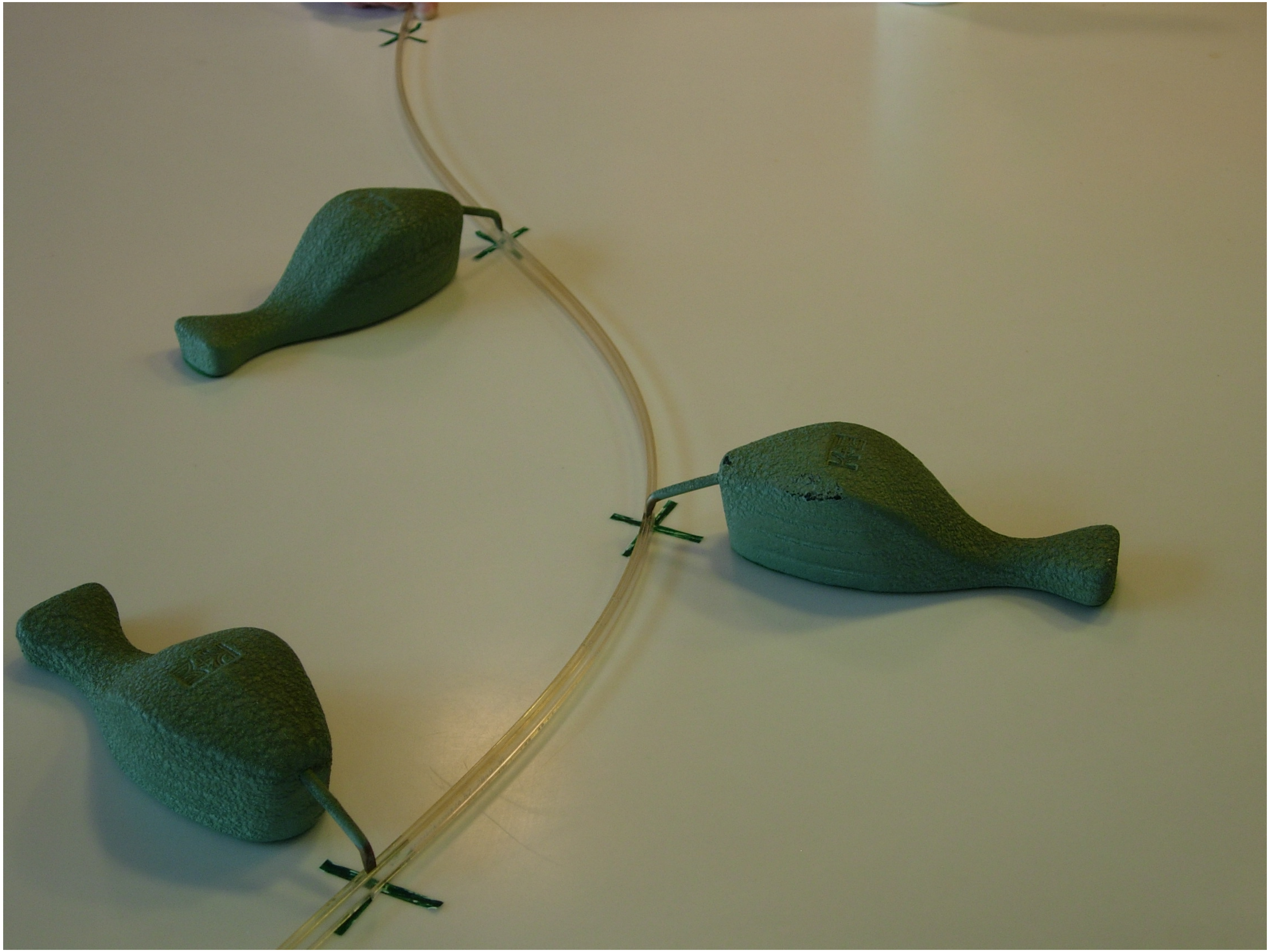
The Chemical Speciation Network are a series of air quality monitors run by EPA (221 locations in 2007). We'll look at a subset of the data from Nov 11th, 2007 (n=191) for just PM2.5.



```r
csn
```

```
# A tibble: 191 × 5
      site longitude latitude date               
     <int>     <dbl>    <dbl> <dttm>             
1 10730023     -86.8     33.6 2007-11-14 00:00:00
2 10732003     -86.9     33.5 2007-11-14 00:00:00
3 10890014     -86.6     34.7 2007-11-14 00:00:00
4 11011002     -86.3     32.4 2007-11-14 00:00:00
5 11130001     -85.0     32.5 2007-11-14 00:00:00
6 40139997    -112.      33.5 2007-11-14 00:00:00
7 40191028    -111.      32.3 2007-11-14 00:00:00
8 51190007     -92.3     34.8 2007-11-14 00:00:00
9 60070002    -122.      39.8 2007-11-14 00:00:00
```

Aside - Splines



Splines in 1d - Smoothing Splines

These are a mathematical analogue to the drafting splines represented using a penalized regression model.

We want to find a function $f(x)$ that best fits our observed data $\boldsymbol{y} = y_1, \ldots, y_n$ while being *smooth*.

$$\underset{f(x)}{\arg\min} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_{-\infty}^{\infty} f''(x)^2 \, dx$$

Interestingly, this minimization problem has an exact solution which is given by a mixture of weighted natural cubic splines (cubic splines that are linear in the tails) with knots at the observed data locations (xs).

Splines in 2d - Thin Plate Splines

Now imagine we have observed data of the form (x_i, y_i, z_i) where we wish to predict z_i given x_i and y_i for all i. We can extend the smoothing spline model in two dimensions,

$$\underset{f(x,y)}{\arg\min} \sum_{i=1}^{n} (z_i - f(x_i, y_i))^2 + \lambda \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left(\frac{\partial^2 f}{\partial x^2} + 2 \frac{\partial^2 f}{\partial x \, \partial y} + \frac{\partial^2 f}{\partial y^2} \right) dx \, dy$$

The solution to this equation has a natural representation using a weighted sum of *radial basis functions* with knots at the observed data locations ($\boldsymbol{x}_i$)

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} w_i \, d(\boldsymbol{x}, \boldsymbol{x}_i)^2 \log d(\boldsymbol{x}, \boldsymbol{x}_i).$$

Prediction locations

```
r_usa = stars::st_rasterize(
  usa,
  stars::st_as_stars(st_bbox(usa),
    nx = 100, ny = 50, values=NA_real_)
)
plot(r_usa)
```


ID

Fitting a TPS

```r
coords = select(csn, long=longitude, lat=latitude) |>
  as.matrix()
(tps = fields::Tps(x = coords, Y=csn$pm25, lon.lat=TRUE))
```

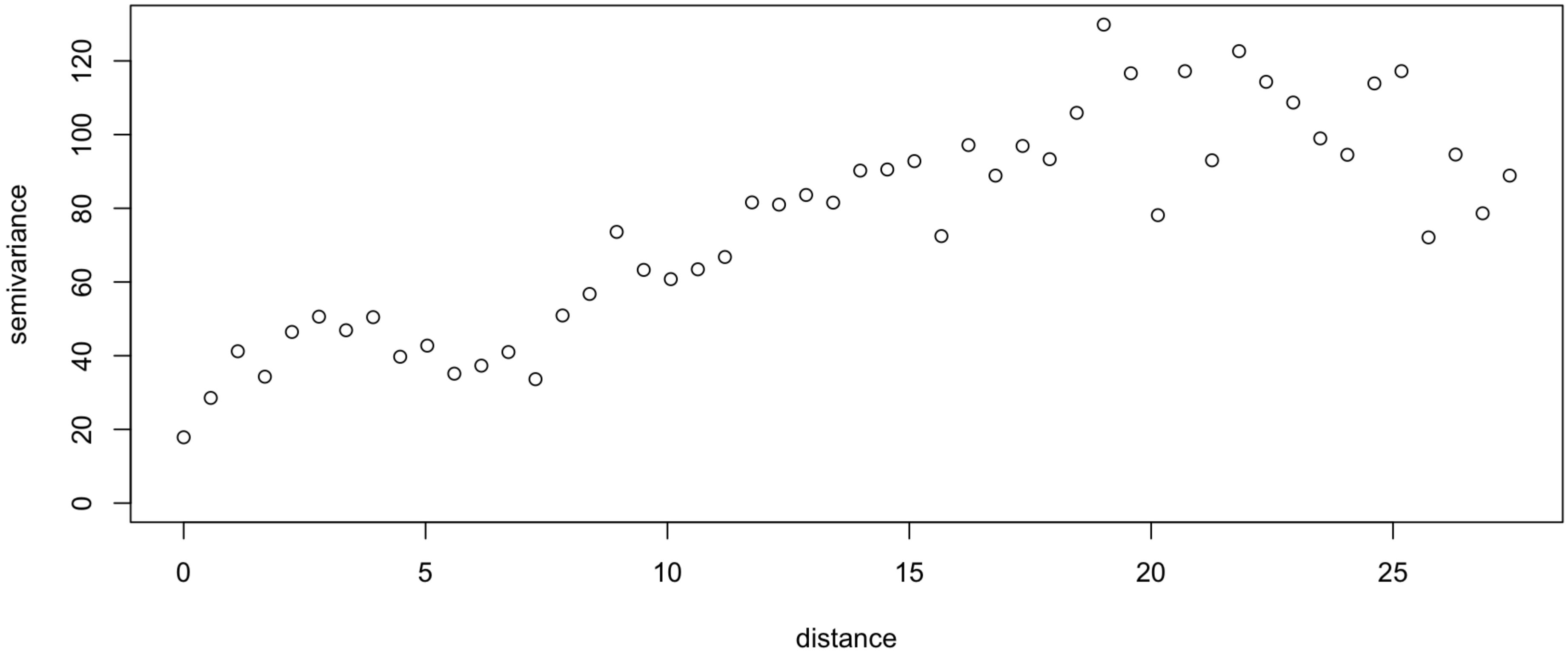
```
Call:
fields::Tps(x = coords, Y = csn$pm25, lon.lat = TRUE)

 Number of Observations:                191
 Number of parameters in the null space 3
 Parameters for fixed spatial drift     3
 Model degrees of freedom:              64
 Residual degrees of freedom:           127
 GCV estimate for tau:                  4.461
 MLE for tau:                           4.286
 MLE for sigma:                         15.35
 lambda                                 1.2
 User supplied sigma                    NA
 User supplied tau^2                    NA
Summary of estimates:
             lambda      trA      GCV    tauHat
```

Predictions

Gaussin Process Models / Kriging

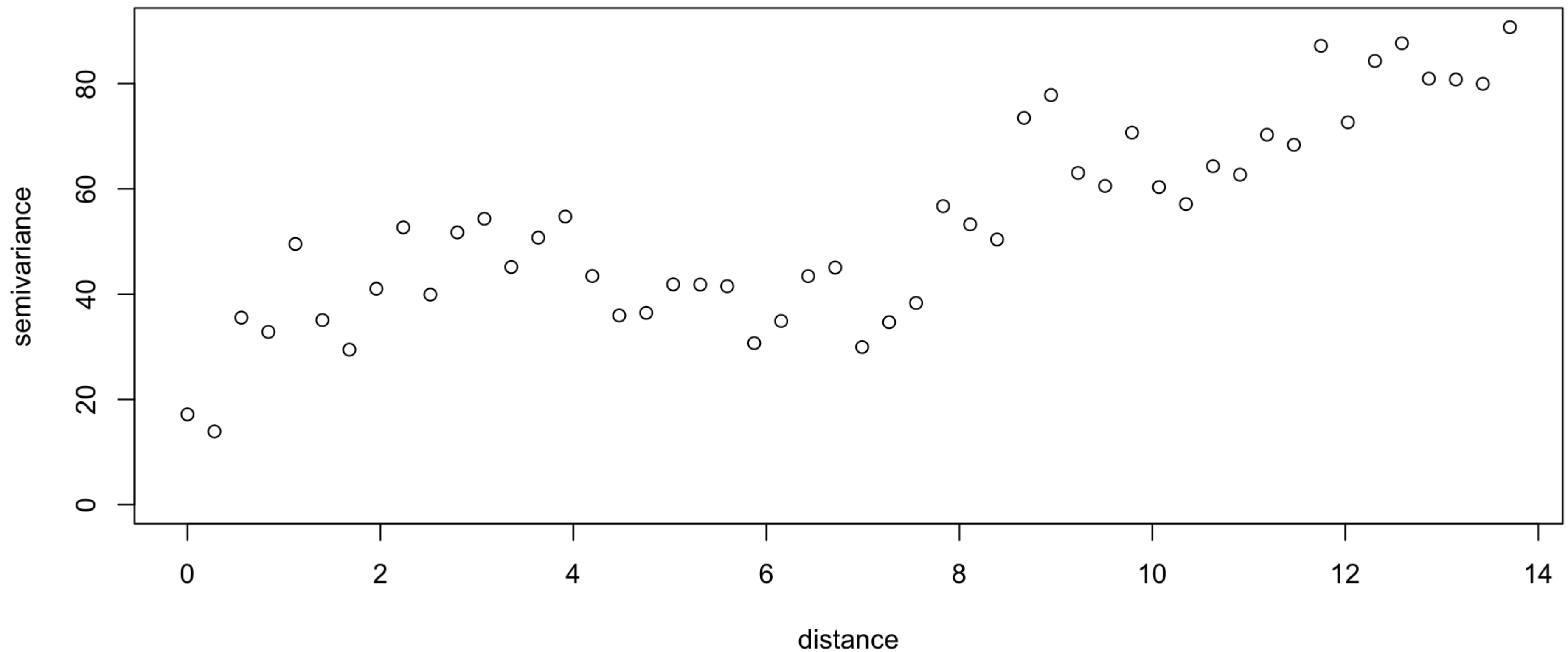
Variogram

```r
coords = csn %>% select(latitude, longitude) %>% as.matrix()
d = fields::rdist(coords)

geoR::variog(
  coords = coords, data = csn$pm25, messages = FALSE,
  uvec = seq(0, max(d)/2, length.out=50)
) %>%
  plot()
```



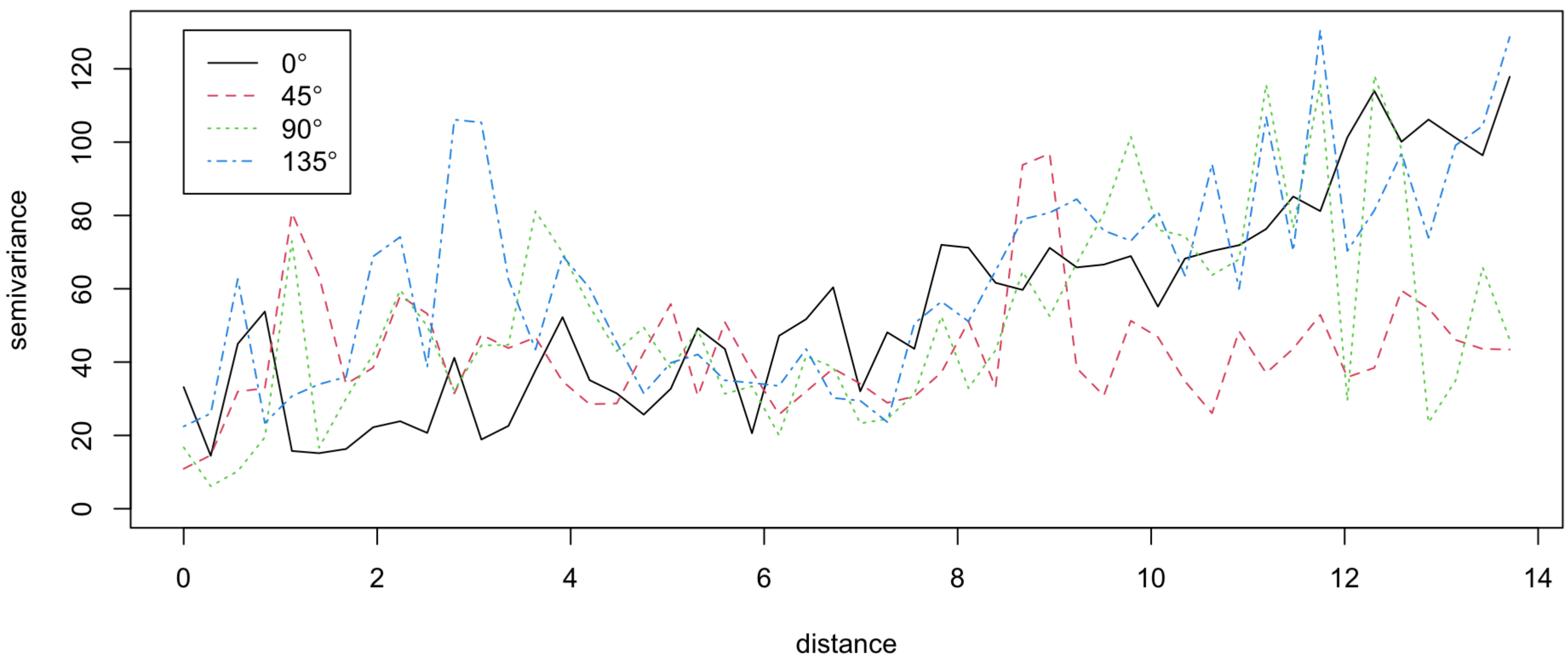
semivariance
0
20
40
60
80
100
120
0
5
10
15
20
25
distance

```r
geoR::variog(
  coords = coords, data = csn$pm25, messages = FALSE,
  uvec = seq(0, max(d)/4, length.out=50)
) %>% plot()
```



Isotropy / Anisotropy

```r
v4 = geoR::variog4(
  coords = coords, data = csn$pm25, messages = FALSE,
  uvec = seq(0, max(d)/4, length.out = 50)
)
plot(v4)
```



GP Spatial Model

If we assume that our data is *stationary* and *isotropic* then we can use a Gaussian Process model to fit the data. We will assume an exponential covariance structure.

$$\boldsymbol{y} \sim \quad (\boldsymbol{\mu},\ \Sigma)$$

$$\{\Sigma\}_{ij} = \sigma^2 \exp(-l \, \|s_i - s_j\|) + \sigma_n^2 \, 1_{i=j}$$

we can also view this as a spatial random effects model where

$$y(\boldsymbol{s}) = \mu(\boldsymbol{s}) + w(\boldsymbol{s}) + \epsilon(\boldsymbol{s})$$

$$w(\boldsymbol{s}) \sim \quad (0, \Sigma')$$

$$\epsilon(s_i) \sim \quad (0, \sigma_n^2)$$

$$\{\Sigma'\}_{ij} = \sigma^2 \exp(-r \, \|s_i - s_j\|)$$

Fitting with `gplm()` (spBayes)

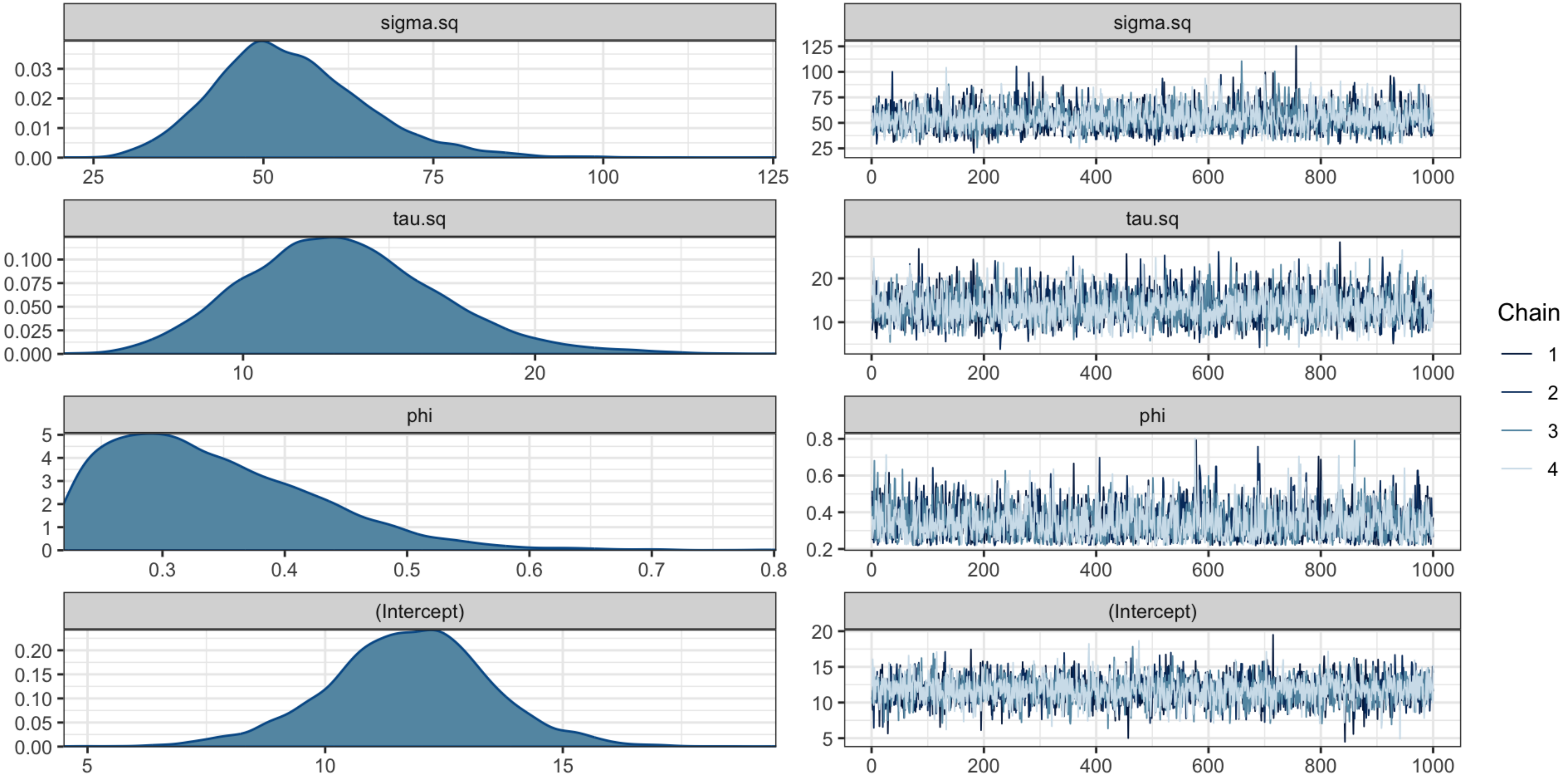
```r
max_range = max(dist(csn[,c("longitude", "latitude")])) / 4

m = gplm(
  pm25~1, data = csn, coords=c("longitude", "latitude"),
  cov_model = "exponential",
  starting = list(phi = 3/3, sigma.sq = 33, tau.sq = 17),
  tuning = list("phi"=0.1, "sigma.sq"=0.1, "tau.sq"=0.1),
  priors = list(
    phi.Unif = c(3/max_range, 3/(0.5)),
    sigma.sq.IG = c(2, 2),
    tau.sq.IG = c(2, 2)
  ),
  thin=10,
  verbose=TRUE
)
```

```
m
```

```
# A gplm model (spBayes spLM) with 4 chains, 4 variables, and 4000
iterations.
# A tibble: 4 × 10
  variable       mean median      sd     mad     q5
  <chr>         <dbl>  <dbl>   <dbl>   <dbl>  <dbl>
1 sigma.sq    54.0   52.9   11.3    10.4    37.7
2 tau.sq      13.4   13.2    3.33    3.20    8.28
3 phi          0.341  0.325  0.0848  0.0835  0.232
4 (Intercept) 11.8   11.8    1.71    1.59    8.87
# … with 4 more variables: q95 <dbl>, rhat <dbl>,
#   ess_bulk <dbl>, ess_tail <dbl>
```

Parameter values

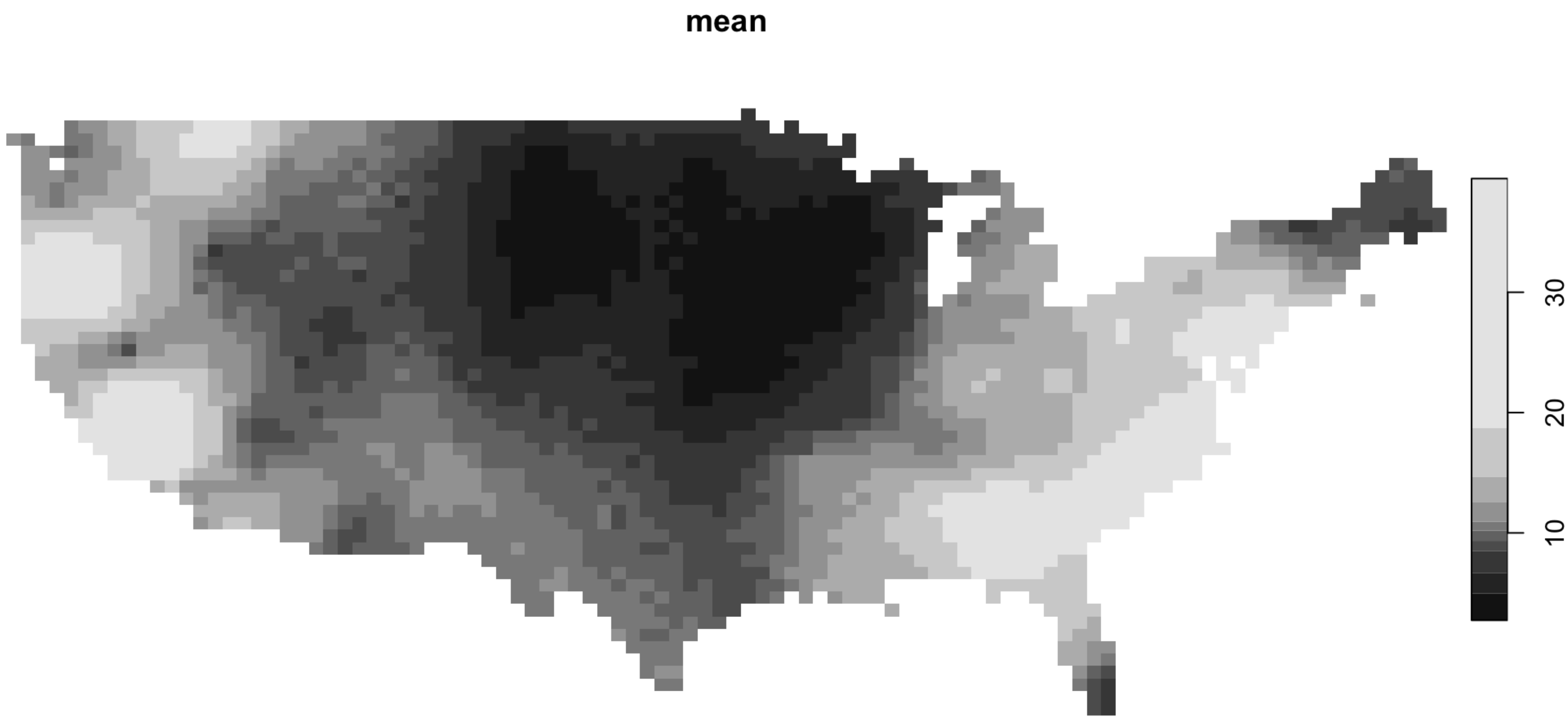
```
plot(m)
```



Predictions

```
(p = predict(m, newdata=pred, coords=c("longitude", "latitude")))
```

```
# A draws_matrix: 1000 iterations, 4 chains, and 2828 variables
    variable
draw  y[1]   y[2] y[3] y[4] y[5]  y[6] y[7] y[8]
  1  14.03 -4.073 15.0  4.8 -8.8  7.84   21  4.9
  2  11.71  0.052 10.2  5.8 11.3 14.58   20 10.7
  3  -3.37 17.307 18.4 20.2 23.7 28.46    9 20.4
  4   7.31  2.500  4.6  7.3 23.7 14.63   15 11.8
  5   0.47 10.014 10.4 17.2 14.6 11.17   10 10.0
  6   7.57 11.004 10.6  9.2 10.6 14.56   23 10.0
  7   7.16  6.791 12.8  5.0 22.4  0.88   16 20.1
  8  16.54  9.611  1.8 23.9 23.9 19.23   38 10.0
  9  16.03  3.135 23.7  1.1 12.4 13.10   34 20.1
```


mean
30
20
10

sd