Statistical Modelling I

HES 505 Fall 2023: Session 22

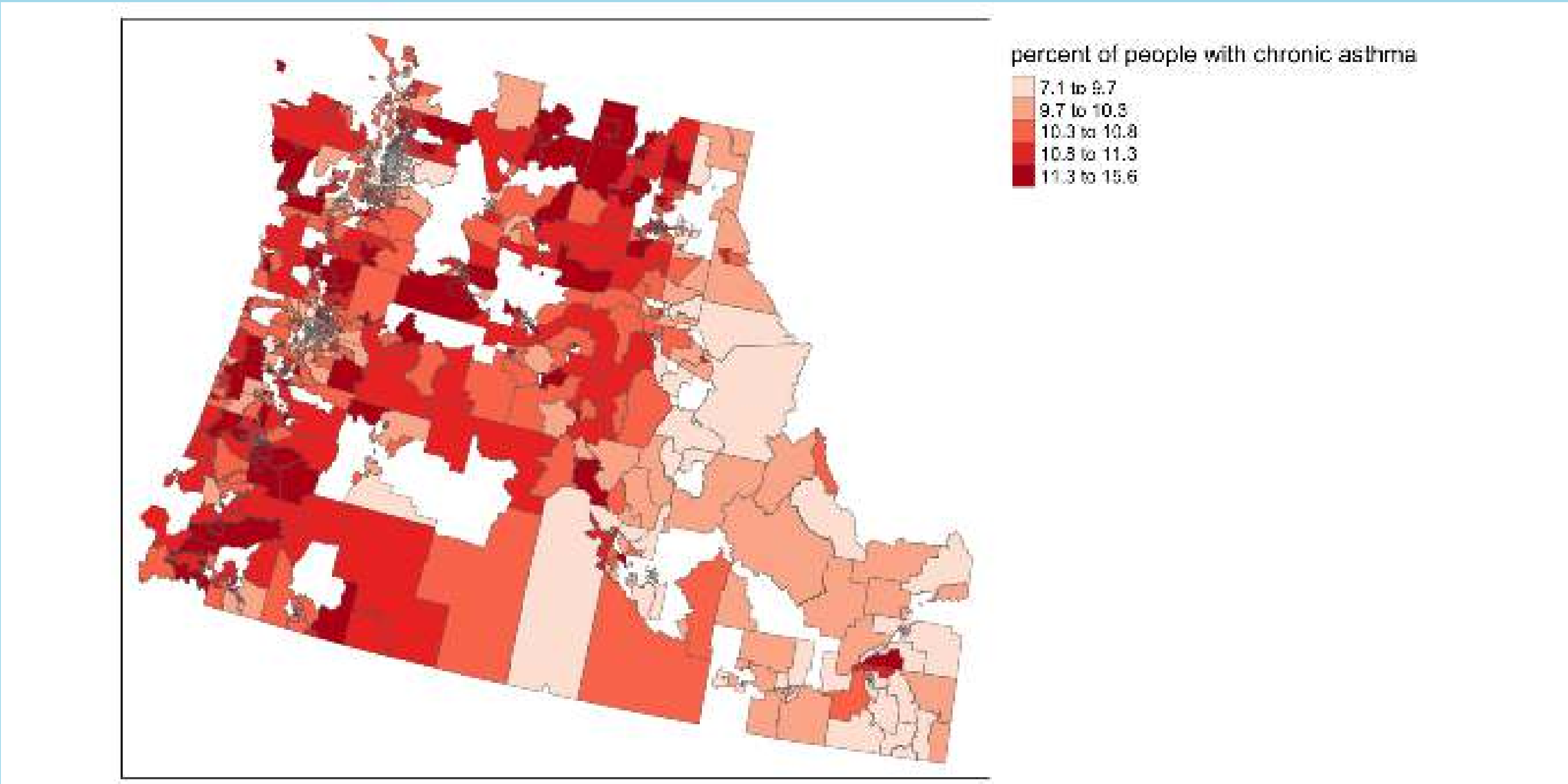
Matt Williamson

Objectives

By the end of today you should be able to:

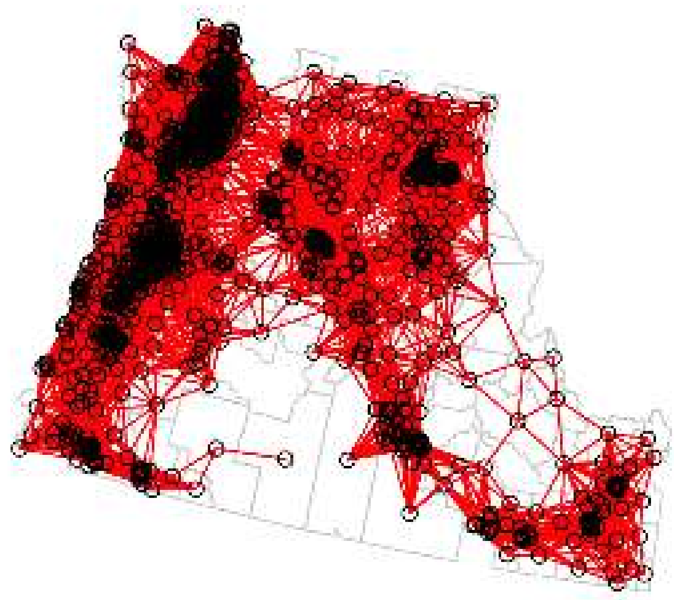
- Identify nearest neighbors based on distance
- Describe and implement overlay analyses
- Extend overlay analysis to statistical modeling
- Generate spatial predictions from statistical models

Revisiting neighbors and areal data




```
cdc.pt <- cdc %>% st_point_on_surface(.)
geog.nearnb <- knn2nb(knearneigh(cdc.pt, k = 1), row.names = cdc.pt$GEOID,
nb.nearest <- dnearneigh(cdc.pt, 0,  max( unlist(nbdists(geog.nearnb, cdc.p
```

Getting Weights



```
lw.nearest <- nb2listw(nb.nearest, style="W")
asthma.lag <- lag.listw(lw.nearest, cdc$casthma_cr)
```

Fit a model

- Moran's I coefficient is the slope of the regression of the *lagged* asthma percentage vs. the asthma percentage in the tract
- More generally it is the slope of the lagged average to the measurement

```
M <- lm(asthma.lag ~ cdc$casthma_cr)
```

```
cdc$casthma_cr
     0.1670774
```

Comparing observed to expected

- We can generate the expected distribution of Moran's I coefficients under a Null hypothesis of no spatial autocorrelation
- Using permutation and a loop to generate simulations of Moran's I

```
n <- 400L   # Define the number of simulations
I.r <- vector(length=n)  # Create an empty vector

for (i in 1:n){
  # Randomly shuffle income values
  x <- sample(cdc$casthma_cr, replace=FALSE)
  # Compute new set of lagged values
  x.lag <- lag.listw(lw.nearest, x)
  # Compute the regression slope and store its value
  M.r    <- lm(x.lag ~ x)
```

```
  I.r[i] <- coef(M.r)[2]
}
```

Significance testing

- Pseudo p-value (based on permutations)
- Analytically (sensitive to deviations from assumptions)
- Using Monte Carlo

```
#Pseudo p-value
N.greater <- sum(coef(M)[2] > I.r)
(p <- min(N.greater + 1, n + 1 - N.greater) / (n + 1))

# Analytically
moran.test(cdc$casthma_cr,lw.nearest, zero.policy = TRUE)

# Monte Carlo
moran.mc(cdc$casthma_cr, lw.nearest, zero.policy = TRUE, nsim=400)
```

Significance testing

```
[1] 0.002493766

	Moran I test under randomisation

data:  cdc$casthma_cr
weights: lw.nearest

Moran I statistic standard deviate = 61.661, p-value < 2.2e-16
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation          Variance
     1.670774e-01     -4.990020e-04      7.385950e-06


	Monte-Carlo simulation of Moran I

data:  cdc$casthma_cr
weights: lw.nearest
number of simulations + 1: 401

statistic = 0.16708, observed rank = 401, p-value = 0.002494
alternative hypothesis: greater
```

Overlay Analyses

Overlays

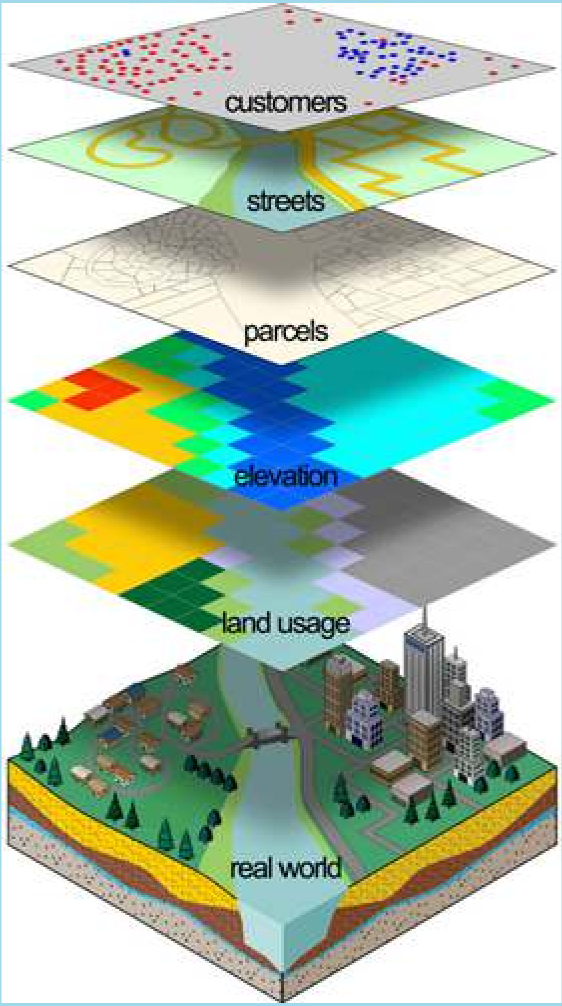
- Methods for identifying optimal site selection or suitability
- Apply a common scale to diverse or disimilar outputs

Getting Started

1. Define the problem.
2. Break the problem into submodels.
3. Determine significant layers.
4. Reclassify or transform the data within a layer.
5. Add or combine the layers.
6. Verify

Boolean Overlays

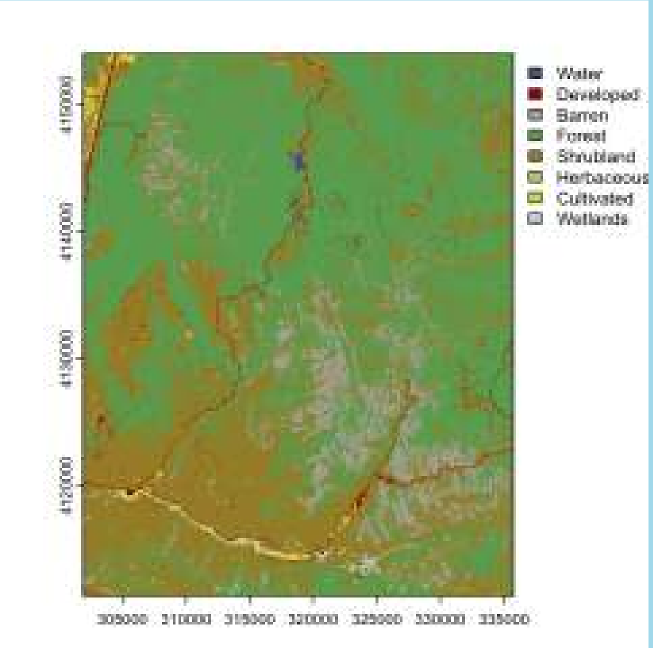
- Successive disqualification of areas
- Series of “yes/no” questions
- “Sieve” mapping



Boolean Overlays

- Reclassifying
- Which types of land are appropriate

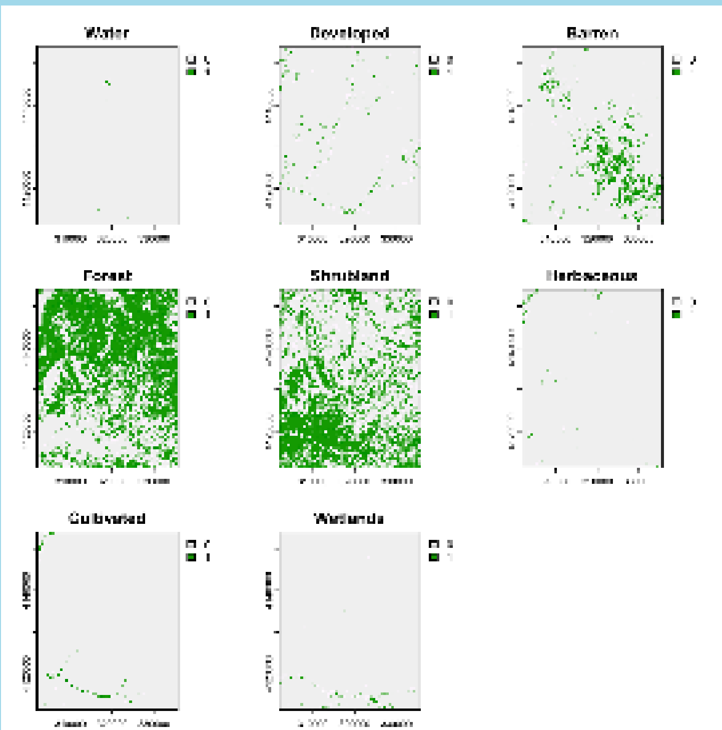
```
nlcd <-  rast(system.file("raster/nlcd.tif", package = "spDataLarge"))
plot(nlcd)
```



Boolean Overlays

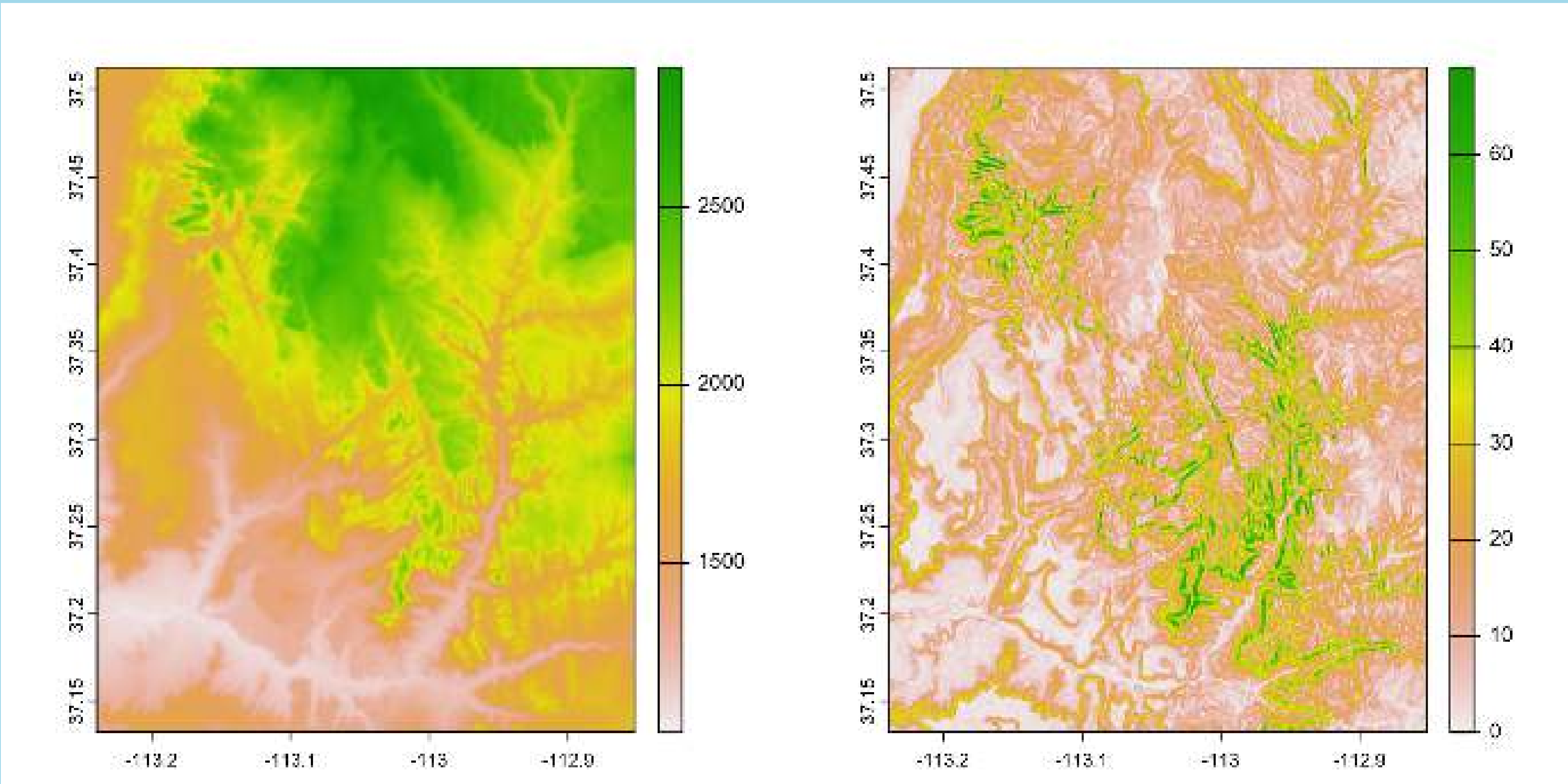
- Which types of land are appropriate?

```r
nlcd.segments <- segregate(nlcd)
names(nlcd.segments) <- levels(nlcd)[[1]][-1,2]
plot(nlcd.segments)
```



Boolean Overlays

- Which types of land are appropriate?

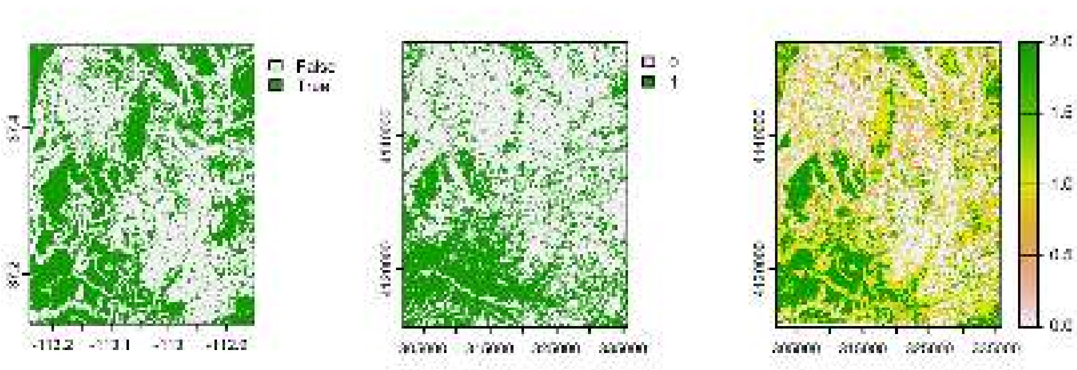


Boolean Overlays

- Make sure data is aligned!

```
suit.slope <- slope < 10
suit.landcov <- nlcd.segments["Shrubland"]
suit.slope.match <- project(suit.slope, suit.landcov)
suit <- suit.slope.match + suit.landcov
```

Boolean Overlays



Challenges with Boolean Overlays

1. Assume relationships are really Boolean
2. No measurement error
3. Categorical measurements are known exactly
4. Boundaries are well-represented

A more general approach

- Define a *favorability* metric

$$F(\mathbf{s}) = \prod_{M=1}^{m} X_m(\mathbf{s})$$

- Treat $F(\mathbf{s})$ as binary
- Then $F(\mathbf{s}) = 1$ if all inputs ($X_m(\mathbf{s})$) are suitable
- Then $F(\mathbf{s}) = 0$ if not

Estimating favorability

$$F(\mathbf{s}) = f(w_1 X_1(\mathbf{s}), w_2 X_2(\mathbf{s}), w_3 X_3(\mathbf{s}), \ldots, w_m X_m(\mathbf{s}))$$

- $F(\mathbf{s})$ does not have to be binary (could be ordinal or continuous)
- $X_m(\mathbf{s})$ could also be extended beyond simply 'suitable / not suitable'
- Adding weights allows incorporation of relative importance
- Other functions for combining inputs ($X_m(\mathbf{s})$)

Weighted Linear Combinations

$$F(\mathbf{s}) = \frac{\sum_{i=1}^{m} w_i X_i(\mathbf{s})}{\sum_{i=1}^{m} w_i}$$

- F(s) is now an index based of the values of $X_m(\mathbf{s})$
- w_i can incorporate weights of evidence, uncertainty, or different participant preferences
- Dividing by $\sum_{i=1}^{m} w_i$ normalizes by the sum of weights

Model-driven overlay

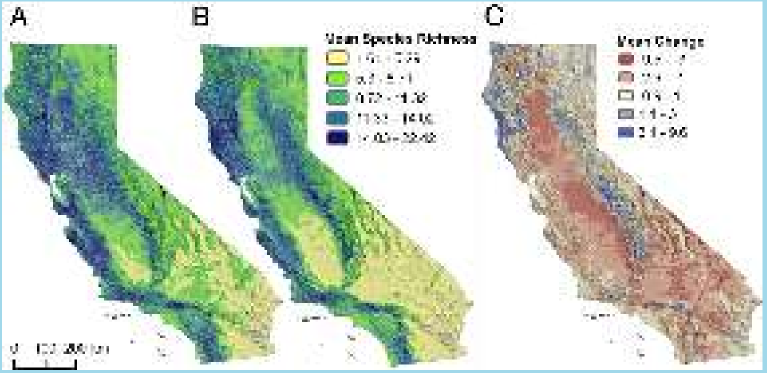
$$F(\mathbf{s}) = w_0 + \sum_{i=1}^{m} w_i X_i(\mathbf{s}) + \epsilon$$

- If we estimate w_i using data, we specify F(s) as the outcome of regression
- When F(s) is binary → logistic regression
- When F(s) is continuous → linear (gamma) regression
- When F(s) is discrete → Poisson regression
- Assumptions about ϵ matter!!

Logistic Regression and Distribution Models

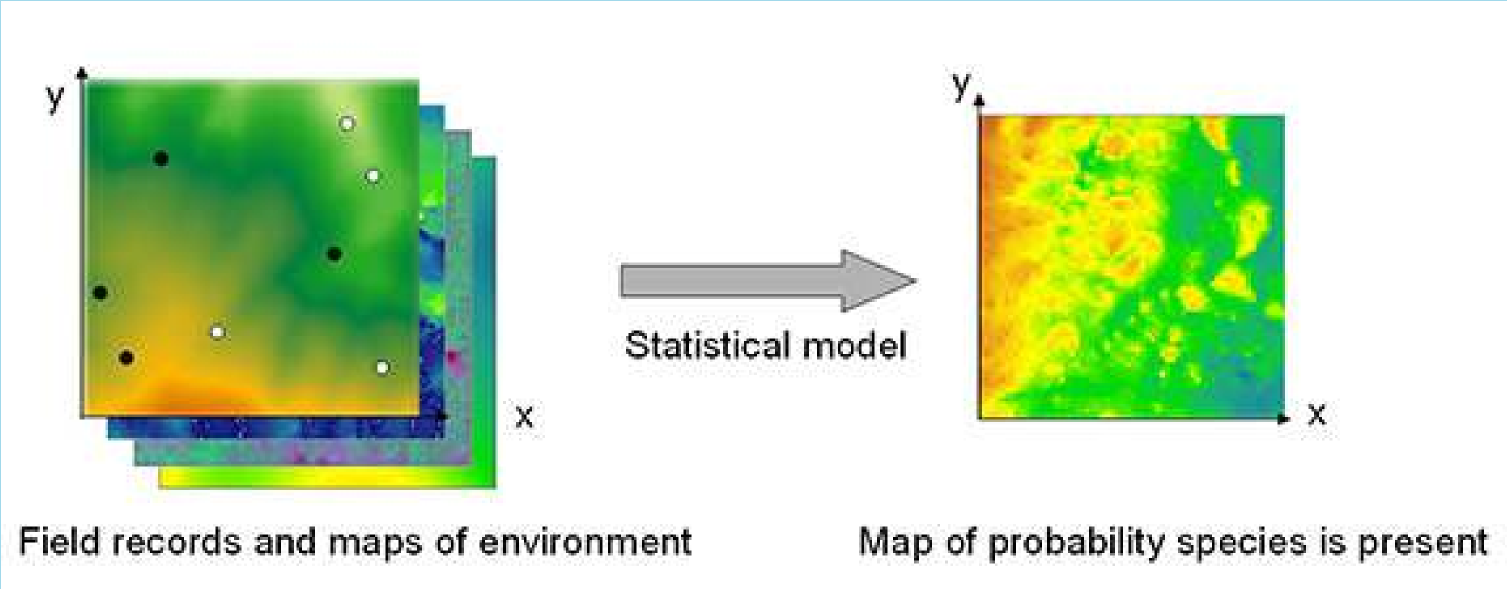
Why do we create distribution models?

- To identify important correlations between predictors and the occurrence of an event
- Generate maps of the 'range' or 'niche' of events
- Understand spatial patterns of event co-occurrence
- Forecast changes in event distributions



From Wiens et al. 2009

General analysis situation



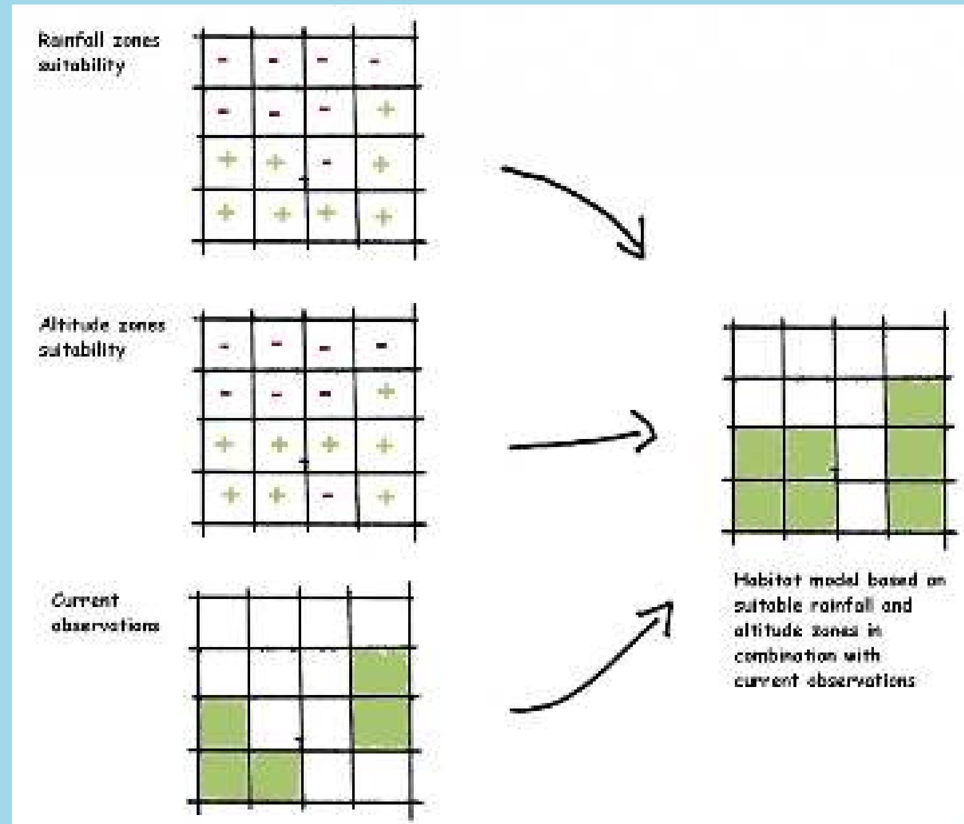
From Long

- Spatially referenced locations of events (**y**) sampled from the study extent
- A matrix of predictors (**X**) that can be assigned to each event based on spatial location

> **Goal**: Estimate the probability of occurrence of events across unsampled regions of the study area based on correlations with predictors

Modeling Presence-Absence Data

- Random or systematic sample of the study region
- The presence (or absence) of the event is recorded for each point
- Hypothesized predictors of occurrence are measured (or extracted) at each point

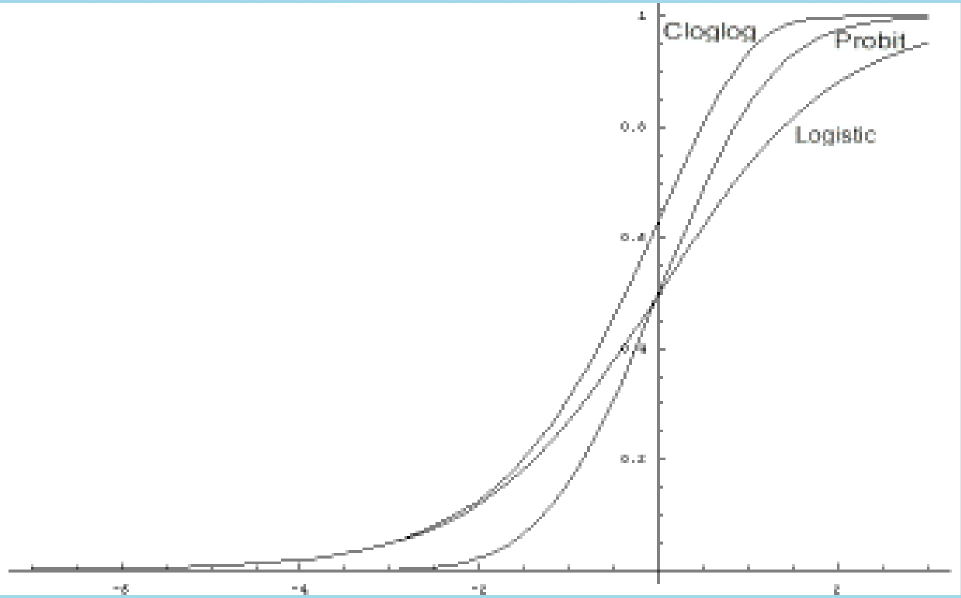


Logistic regression

- We can model favorability as the **probability** of occurrence using a logistic regression
- A *link* function maps the linear predictor ($\mathbf{x_i}'\beta + \alpha$) onto the support (0-1) for probabilities
- Estimates of β can then be used to generate 'wall-to-wall' spatial predictions

$$y_i \sim \mathrm{Bern}(p_i)$$

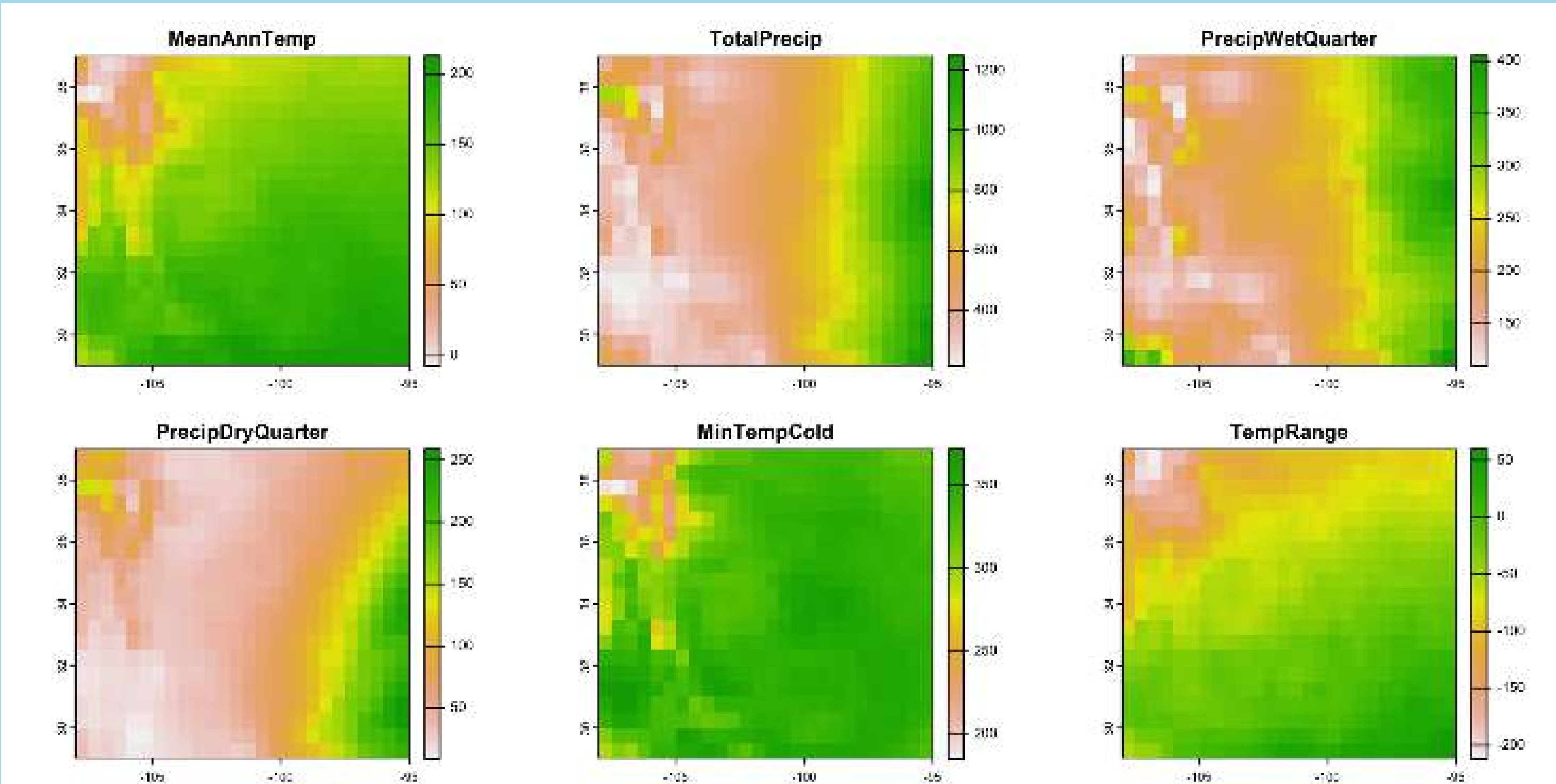
$$\mathrm{link}(p_i) = \mathbf{x_i}'\beta + \alpha$$



From Mendoza

An Example

Inputs from the `dismo` package

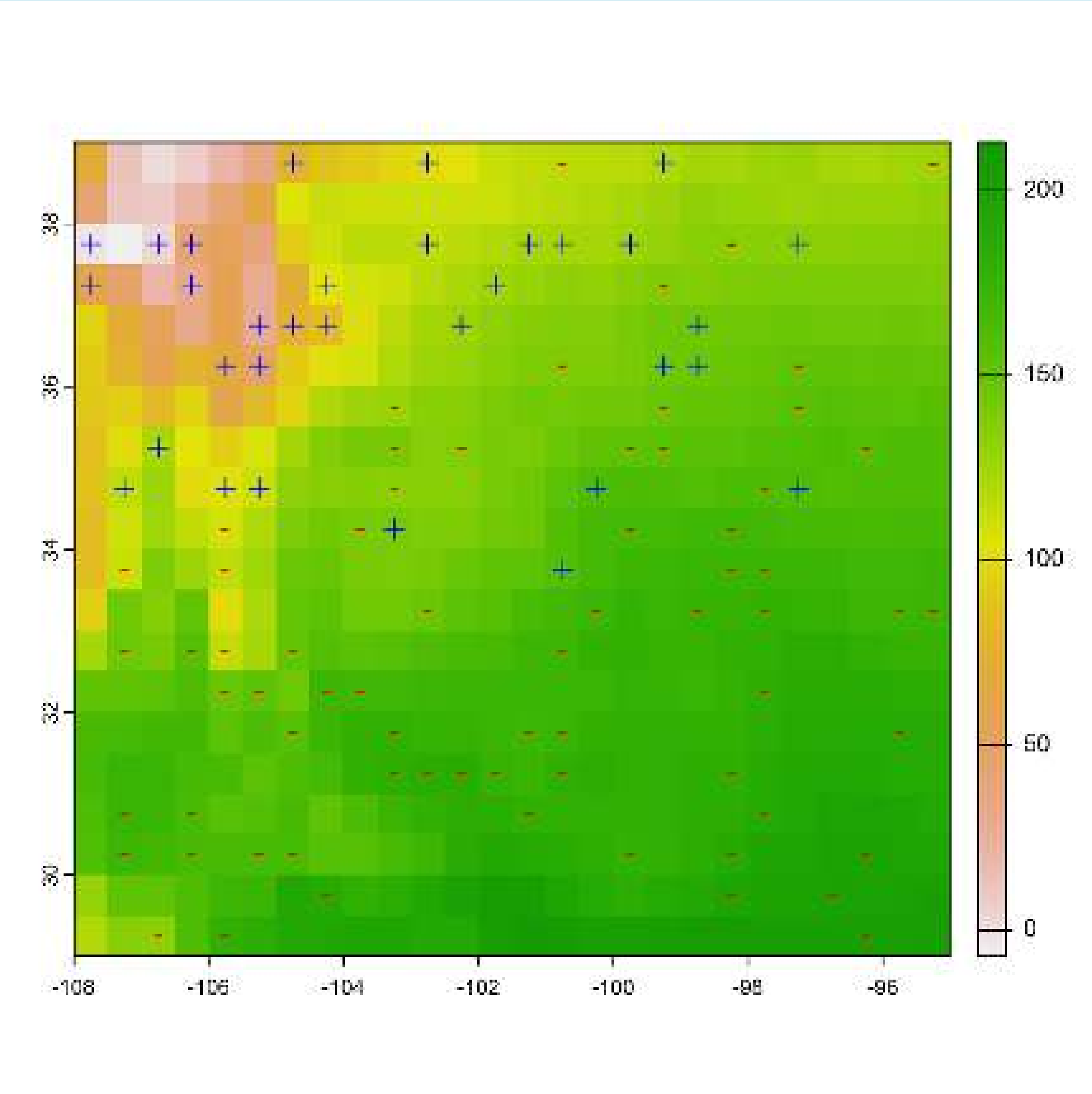


An Example

The sample data

```r
head(pres.abs)
```

```
Simple feature collection
with 6 features and 1 field
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: -106.75
ymin: 31.25 xmax: -98.75
ymax: 37.75
Geodetic CRS:  GCS_unknown
  y                geometry
1 0  POINT (-99.25 35.25)
2 1  POINT (-98.75 36.25)
3 1 POINT (-106.75 35.25)
4 0 POINT (-100.75 31.25)
5 1  POINT (-99.75 37.75)
6 1 POINT (-104.25 36.75)
```



An Example

Building our dataframe

```r
pts.df <- terra::extract(pred.stack, vect(pres.abs), df=TRUE)
head(pts.df)
```

```
  ID MeanAnnTemp TotalPrecip PrecipWetQuarter PrecipDryQuarter MinTempCold
1  1         155         667              253               71         350
2  2         147         678              266               66         351
3  3         123         261              117               40         329
4  4         181         533              198               69         348
5  5         127         589              257               48         338
6  6          83         438              213               38         278
  TempRange
1       -45
2       -58
3       -64
4        -5
5       -81
6      -107
```

An Example

Building our dataframe

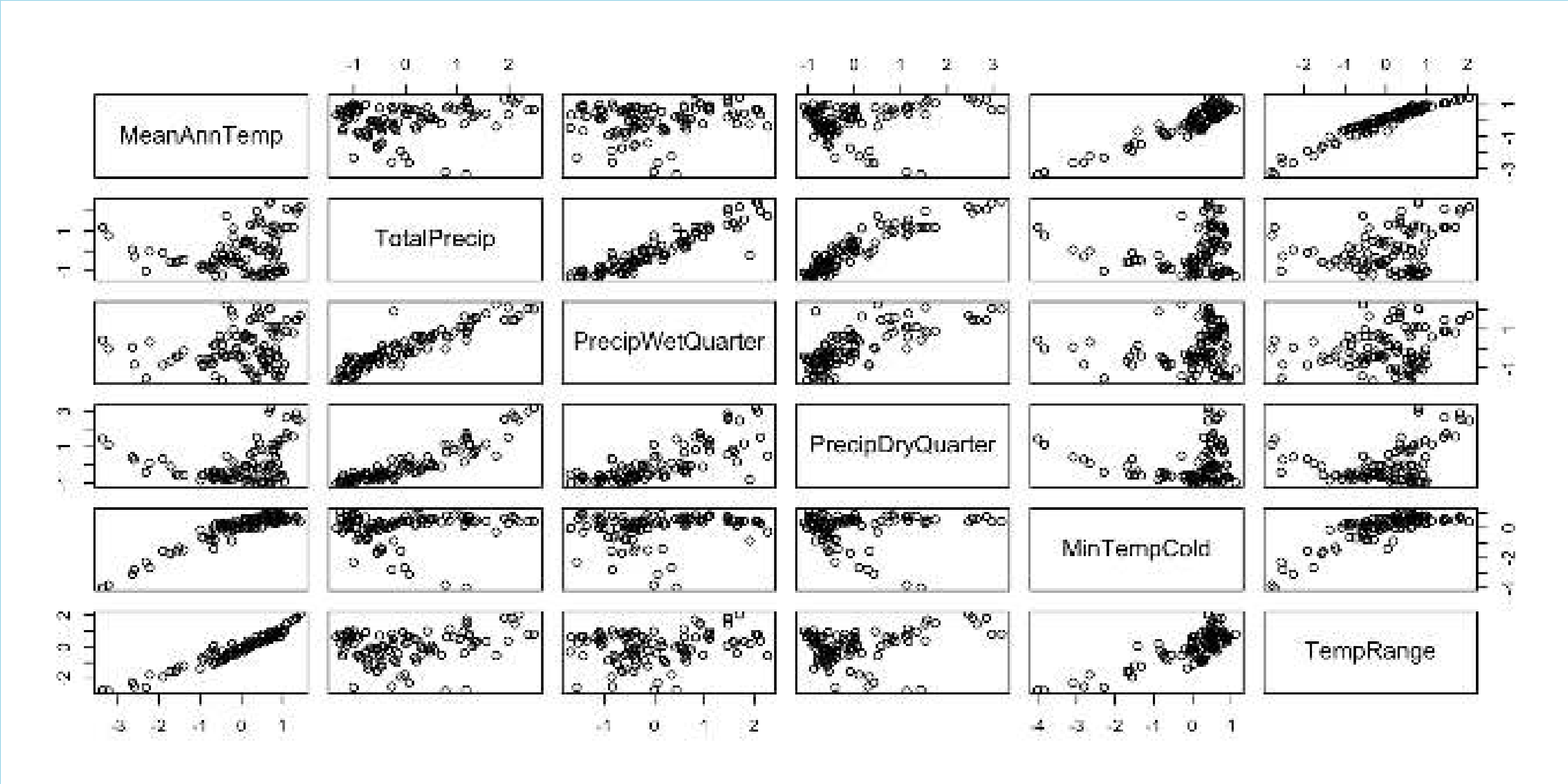
```r
pts.df[,2:7] <- scale(pts.df[,2:7])
summary(pts.df)
```

```
       ID          MeanAnnTemp        TotalPrecip       PrecipWetQuarter
 Min.   :  1.00   Min.   :-3.3729   Min.   :-1.3377   Min.   :-1.6926
 1st Qu.: 25.75   1st Qu.:-0.4594   1st Qu.:-0.7980   1st Qu.:-0.6895
 Median : 50.50   Median : 0.2282   Median :-0.2373   Median :-0.2224
 Mean   : 50.50   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 75.25   3rd Qu.: 0.7118   3rd Qu.: 0.7140   3rd Qu.: 0.6508
 Max.   :100.00   Max.   : 1.4285   Max.   : 2.4843   Max.   : 2.2713
 PrecipDryQuarter   MinTempCold         TempRange
 Min.   :-1.0828   Min.   :-3.9919   Min.   :-2.7924
 1st Qu.:-0.7013   1st Qu.:-0.0598   1st Qu.:-0.5216
 Median :-0.3770   Median : 0.3582   Median : 0.2075
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.4290   3rd Qu.: 0.5495   3rd Qu.: 0.6450
 Max.   : 3.1713   Max.   : 1.1092   Max.   : 2.0407
```

An Example

Looking at correlations

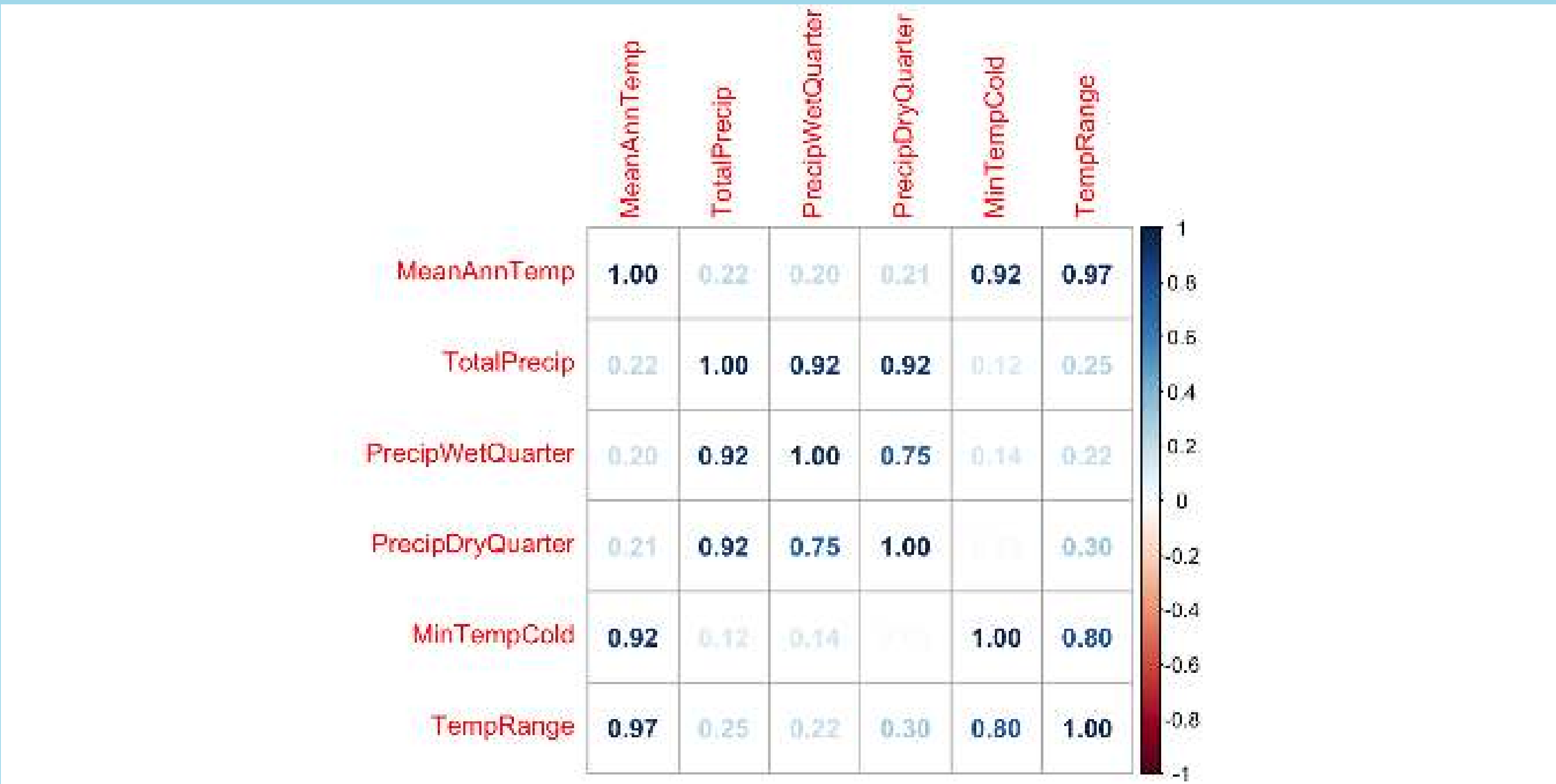
```
1 pairs(pts.df[,2:7])
```



An Example

Looking at correlations

```r
corrplot(cor(pts.df[,2:7]), method = "number")
```



An Example

Fitting some models

```
pts.df <- cbind(pts.df, pres.abs$y)
colnames(pts.df)[8] <- "y"
logistic.global <- glm(y~., family=binomial(link="logit"), data=pts.df[,2:8
logistic.simple <- glm(y ~ MeanAnnTemp + TotalPrecip, family=binomial(link=
logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
```

An Example

Checking out the results

```r
summary(logistic.global)
```

```
Call:
glm(formula = y ~ ., family = binomial(link = "logit"), data = pts.df[,
    2:8])

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       -1.4461     0.5096  -2.837  0.00455 **
MeanAnnTemp       -6.3578     6.1645  -1.031  0.30237
TotalPrecip        7.1453     4.5577   1.568  0.11694
PrecipWetQuarter  -5.4207     3.0432  -1.781  0.07487 .
PrecipDryQuarter  -1.3110     2.2482  -0.583  0.55981
MinTempCold        3.0890     2.6334   1.173  0.24080
TempRange         -0.6213     4.5470  -0.137  0.89131
---
```

An Example

Checking out the results

```r
summary(logistic.simple)
```

```
Call:
glm(formula = y ~ MeanAnnTemp + TotalPrecip, family = binomial(link =
"logit"),
    data = pts.df[, 2:8])

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.9880     0.3145  -3.141  0.00168 **
MeanAnnTemp  -2.9990     0.6647  -4.512 6.42e-06 ***
TotalPrecip   0.3924     0.3827   1.025  0.30517
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

An Example

Checking out the results

```r
summary(logistic.rich)
```

```

Call:
glm(formula = y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
    family = binomial(link = "logit"), data = pts.df[, 2:8])

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -0.96504    0.35650  -2.707  0.00679 **
MeanAnnTemp      -2.85446    0.66142  -4.316 1.59e-05 ***
PrecipWetQuarter  0.03212    0.43102   0.075  0.94060
PrecipDryQuarter  0.16759    0.64935   0.258  0.79634
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

An Example

Comparing models

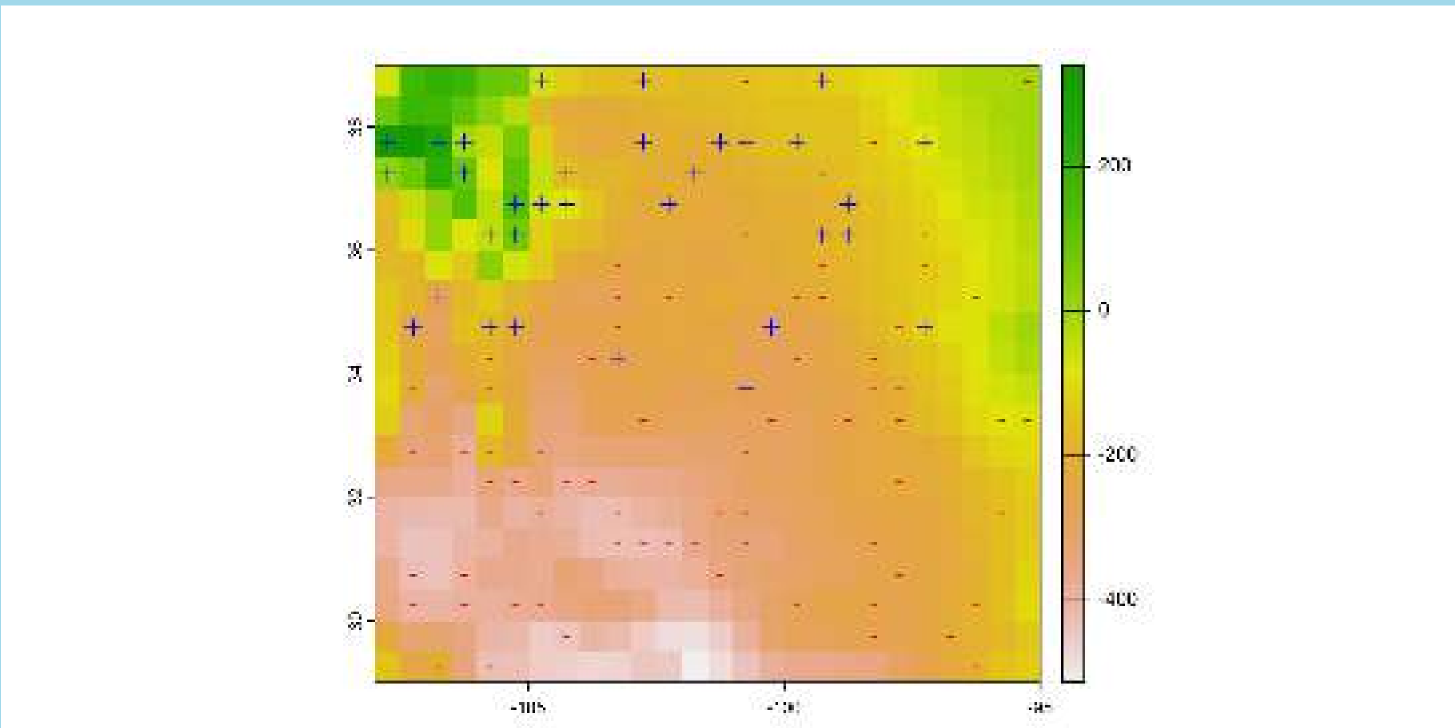
```r
AIC(logistic.global, logistic.simple, logistic.rich)
```

```
                df      AIC
logistic.global  7 65.76394
logistic.simple  3 74.10760
logistic.rich    4 77.00622
```

An Example

Generating predictions

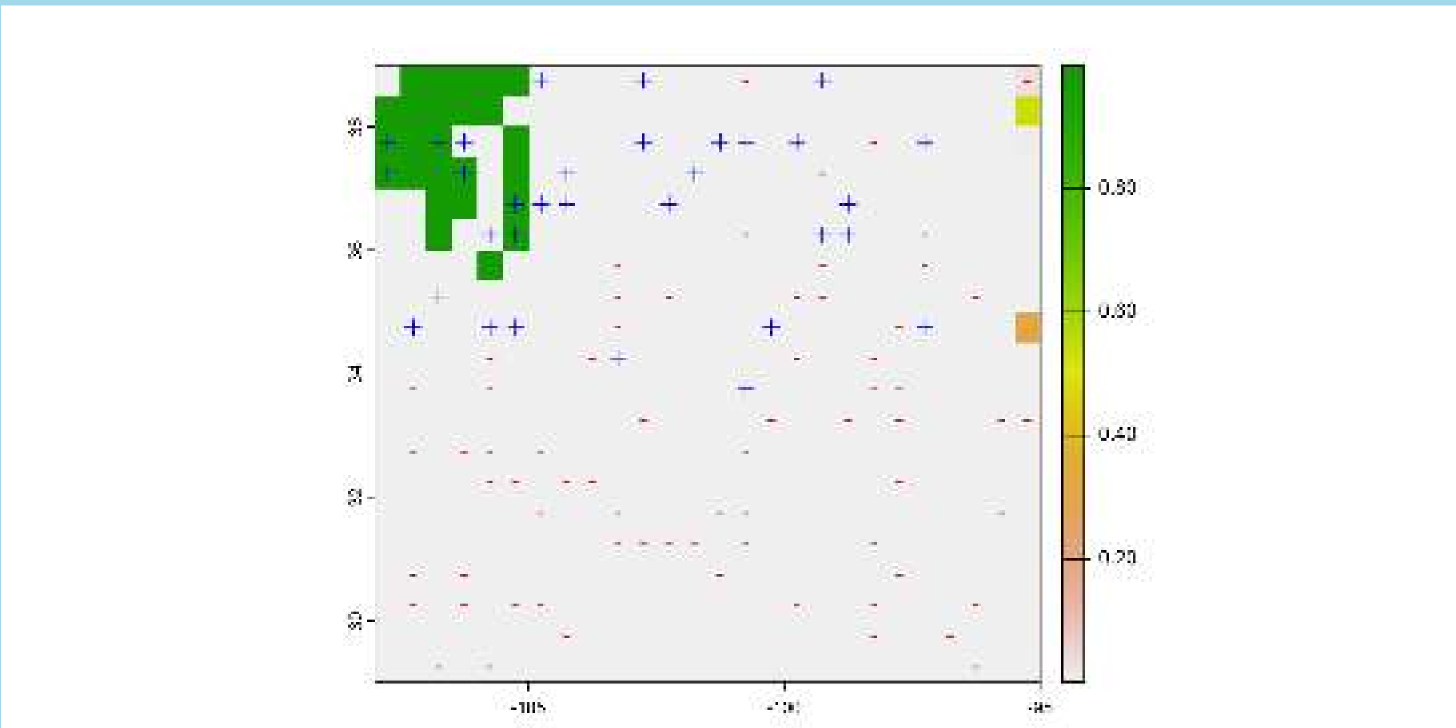
```
preds <- predict(object=pred.stack, model=logistic.simple)
plot(preds)
plot(pres.pts$geometry, add=TRUE, pch=3, col="blue")
plot(abs.pts$geometry, add=TRUE, pch ="-", col="red")
```



An Example

Generating predictions

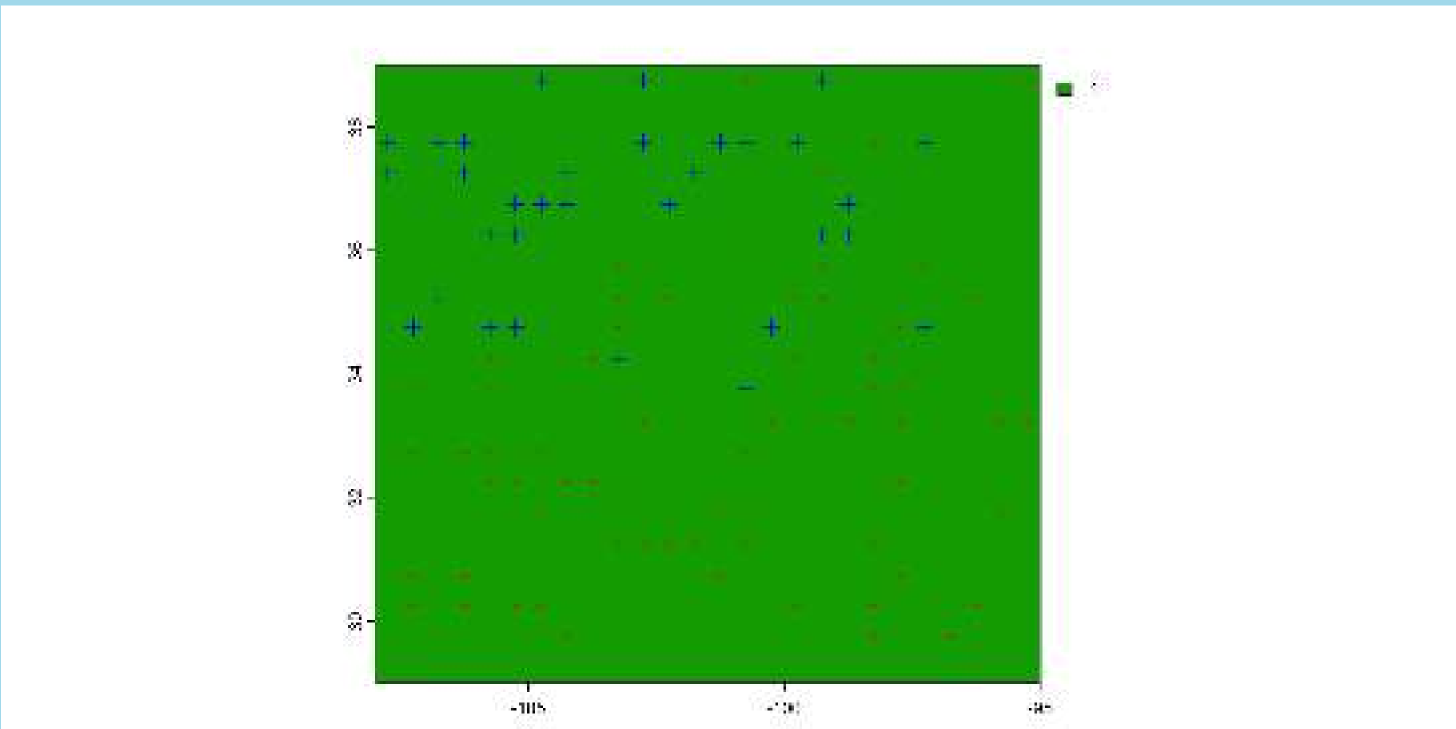
```r
preds <- predict(object=pred.stack, model=logistic.simple, type="response")
plot(preds)
plot(pres.pts$geometry, add=TRUE, pch=3, col="blue")
plot(abs.pts$geometry, add=TRUE, pch ="-", col="red")
```



An Example

Generating predictions

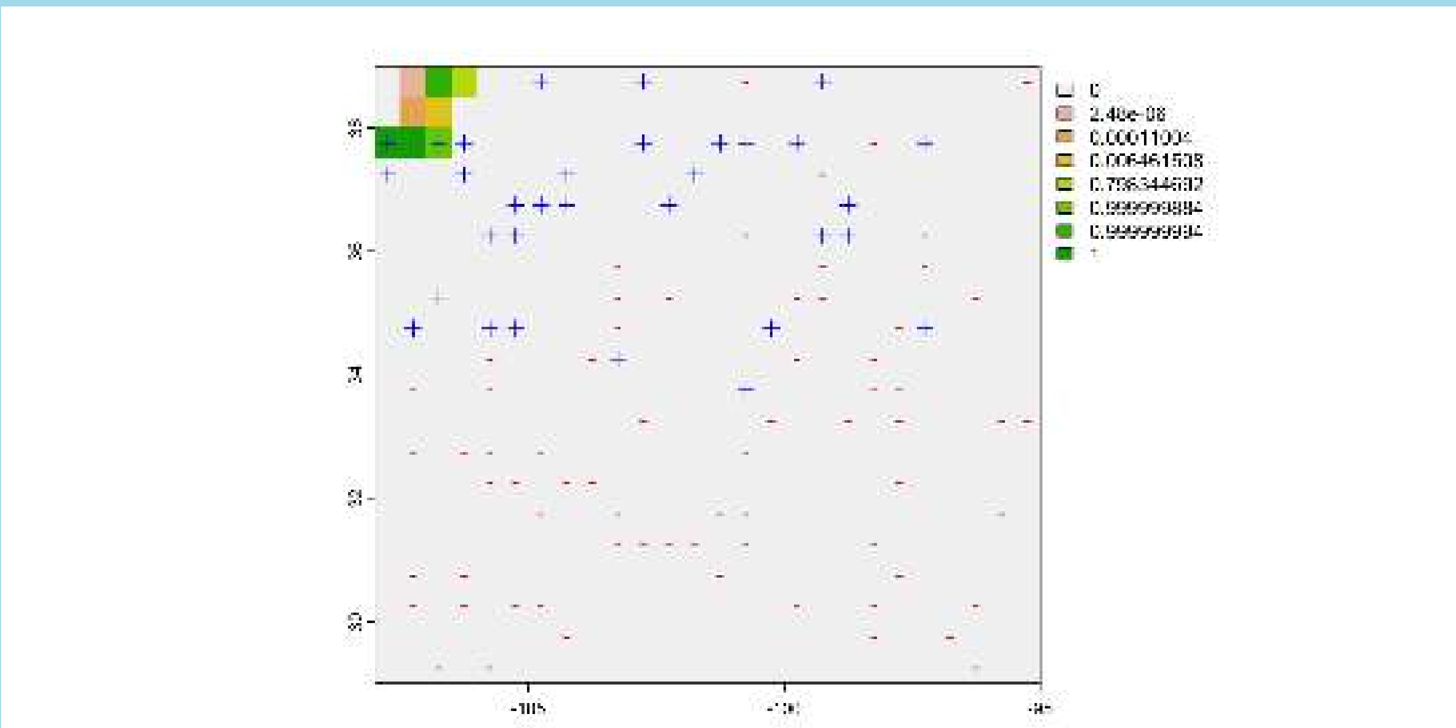
```r
preds <- predict(object=pred.stack, model=logistic.global, type="response")
plot(preds)
plot(pres.pts$geometry, add=TRUE, pch=3, col="blue")
plot(abs.pts$geometry, add=TRUE, pch ="-", col="red")
```



An Example

Generating predictions

```
preds <- predict(object=pred.stack, model=logistic.rich, type="response")
plot(preds)
plot(pres.pts$geometry, add=TRUE, pch=3, col="blue")
plot(abs.pts$geometry, add=TRUE, pch ="-", col="red")
```

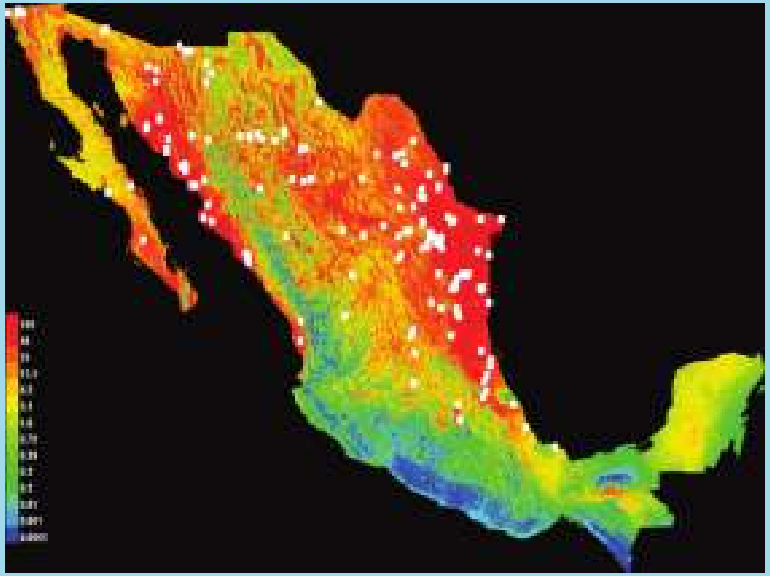


Key assumptions of logistic regression

- Dependent variable must be binary
- Observations must be independent (important for spatial analyses)
- Predictors should not be collinear
- Predictors should be linearly related to the log-odds
- **Sample Size**

Modelling Presence-Background Data

The sampling situation



From Lentz et al. 2008

- Opportunistic collection of presences only
- Hypothesized predictors of occurrence are measured (or extracted) at each presence
- Background points (or pseudoabsences) generated for comparison

The Challenge with Background Points

- What constitutes background?
- Not measuring *probability*, but relative likelihood of occurrence
- Sampling bias affects estimation
- The intercept

$$y_i \sim \mathrm{Bern}(p_i)$$
$$\mathrm{link}(p_i) = \mathbf{x_i}'\beta + \alpha$$