Spatial Statistics in R: An Introduction

John Myles White

February 4, 2010

Caveat Emptor

- I'm not a statistician: I'm a neuroscientist.

Caveat Emptor

- I'm not a statistician: I'm a neuroscientist.
- Spatial statistics is an enormous field, and I can only offer a very broad and general survey of its tools.

Spatial Statistics: A Definition

- Tools for understanding data distributed in a space where positions and distance have meaning.

Spatial Statistics: Example Applications

Examples freely taken from Yuri Zhukov at Harvard:

- How do we model the spread of a contagious disease?

Spatial Statistics: Example Applications

Examples freely taken from Yuri Zhukov at Harvard:

- How do we model the spread of a contagious disease?
- How do we identify crime hot spots?

Spatial Statistics: Example Applications

Examples freely taken from Yuri Zhukov at Harvard:

- How do we model the spread of a contagious disease?
- How do we identify crime hot spots?
- How do we predict housing prices?

Spatial Statistics: What Tools Exist?

- Tools for visualizing spatially distributed data intuitively.

Spatial Statistics: What Tools Exist?

- Tools for visualizing spatially distributed data intuitively.
- Tools for coping with complex coordinate systems.

Spatial Statistics: What Tools Exist?

- Tools for visualizing spatially distributed data intuitively.
- Tools for coping with complex coordinate systems.
- Tools for performing statistical inference on stochastic processes that emit observations in space.

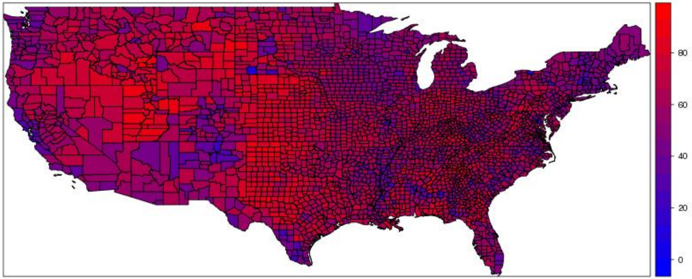
Spatial Statistics: What Tools Exist?

- Tools for visualizing spatially distributed data intuitively.
- Tools for coping with complex coordinate systems.
- Tools for performing statistical inference on stochastic processes that emit observations in space.
- Tools for coping with data sets where the IID assumption is plainly false.

Spatial Data Visualization

From childhood on, we're taught how to read maps. When our data is concerned with counties, states, or nations, maps provide an intuitive way to visualize our data.

A Target Image



What Do We Need to Build Such an Image?

Primitive visual objects that we can combine into larger ensembles:

- Points

What Do We Need to Build Such an Image?

Primitive visual objects that we can combine into larger ensembles:

- Points
- Lines

What Do We Need to Build Such an Image?

Primitive visual objects that we can combine into larger ensembles:

- Points
- Lines
- Polygons

What Do We Need to Build Such an Image?

Primitive visual objects that we can combine into larger ensembles:

- Points
- Lines
- Polygons
- Grids

The Meuse Dataset - Our Spatial Data Example

To illustrate these primitives, we're going to work with data on the concentration of zinc around the Meuse river bank. This data comes from the spatial data package 'sp'.

The Meuse River

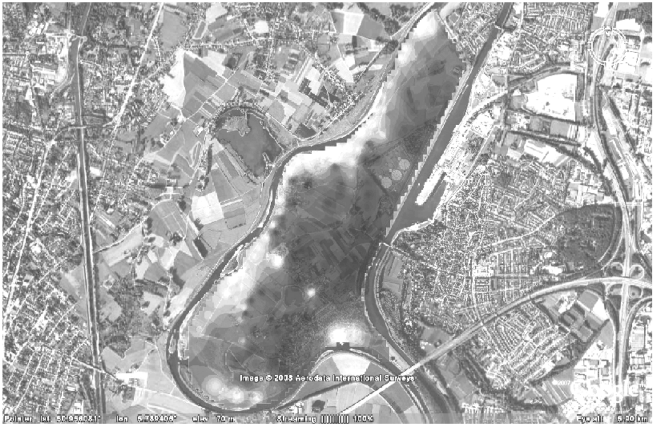


Image © 2008 Aerodata International Surveys
Pointer lat 50.956081° lon 5.739404° elev 70 m
Streaming |||||||| 100%
Eye alt 5.00 km

Working with the Meuse Dataset

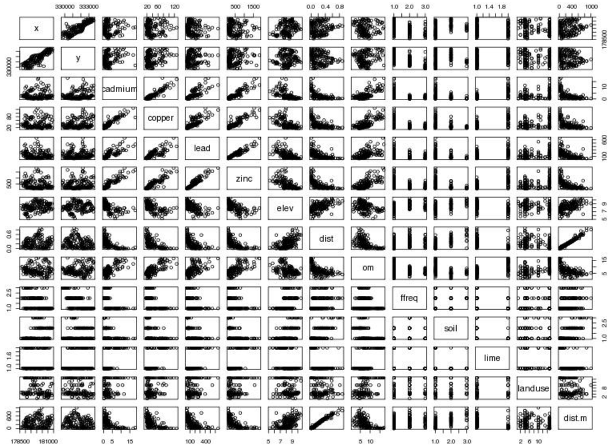
```
> data(meuse)
> class(meuse)
[1] "data.frame"
> head(meuse, n = 1)
       x      y cadmium copper lead zinc
1 181072 333611    11.7     85  299 1022

   elev       dist   om ffreq
  7.909 0.00135803 13.6     1

  soil lime landuse dist.m
1    1    1      Ah     50
```

A Normal R Data Frame

The Meuse dataset starts as a normal R data frame. Calling plot() produces a standard column-by-column scatterplot:

```
> plot(meuse)
```

R's Standard Column-by-Column Scatterplot



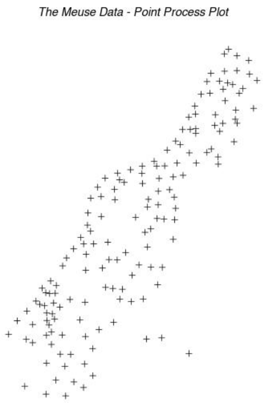
Creating a SpatialPointsDataFrame

Because the Meuse dataset has ‘x’ and ‘y’ columns, we can convert it into a SpatialPointsDataFrame that allows for easy visualization. A SpatialPointsDataFrame is like a standard data frame with additional information that defines the coordinate system for all of the points in the dataset.

The Conversion Process

```
> coordinates(meuse) <- c('x', 'y')
> class(meuse)
[1] "SpatialPointsDataFrame"
attr(,"package")
[1] "sp"
> plot(meuse)
```

Spatial Point Data Frame Plot



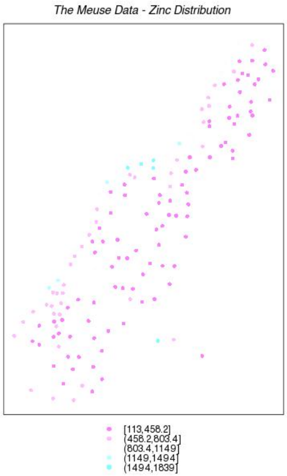
Locations Alone or Locations with Observed Data?

This sort of plot only provides location information. To add another dimension, we use spplot():

Examining Data at These Spatial Points

```
> names(meuse)
[1] "cadmium" "copper"  "lead"    "zinc"
[5] "elev"    "dist"   "om"      "ffreq"   "soil"
[10] "lime"    "landuse" "dist.m"

> spplot(meuse, 'zinc')
```

Zinc Distribution in Space

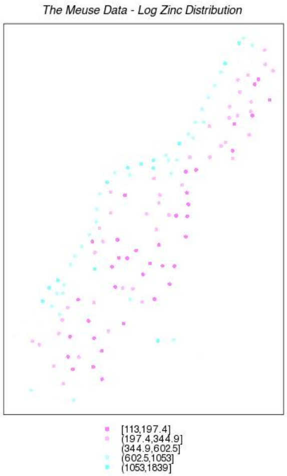


Examining Zinc More Carefully

It turns out that working on a log scale is better for this example:

```
> spplot(meuse, 'zinc', do.log = TRUE)
```

Log Zinc Distribution in Space

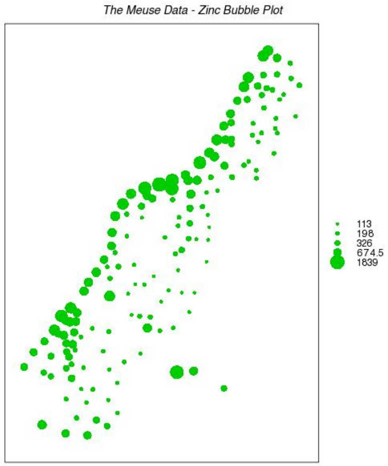


Bubbles: An Alternative Visualization Style

You can also use bubbles to visualize the same data, which I find clearer:

```
> bubble(meuse, 'zinc')
```

Zinc Distribution in Space with Bubbles



Quick Recap

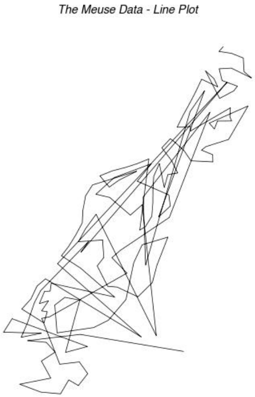
A standard data frame with 2D spatial coordinates can be converted into a ‘SpatialPointsDataFrame’ easily using the ‘coordinates’ function. Then it can be visualized using tools provided by the ‘sp’ package.

Spatial Lines

Beyond points, we also want to have access to lines and polygons:

```
> cc <- coordinates(meuse)
> m.sl <- SpatialLines(list(Lines(list(Line(cc)))))
> plot(m.sl)
```

Spatial Line Plot



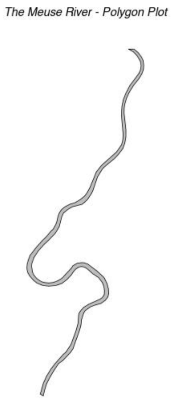
Polygons are More Interesting than Lines

Making lines out of points is clearly not ideal. Let's move on to some polygons that define the Meuse river's shape itself.

Polygons are More Interesting than Lines

```
> data(meuse.riv)
> meuse.lst <- list(Polygons(list(Polygon(meuse.riv)),
     'meuse.riv'))
> meuse.sr <- SpatialPolygons(meuse.lst)
> plot(meuse.sr, col = 'grey')
```

Spatial Polygon Plot



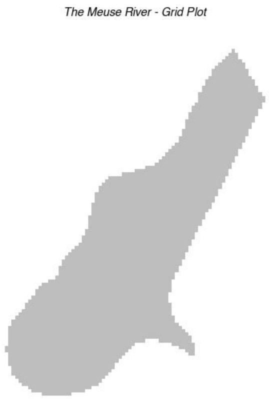
Grid Plots

Beyond the shape of the river, we might want to know something about the location of water around the river bed. For that, we can use a grid:

Grid Plots

```
> data(meuse.grid)
> coordinates(meuse.grid) <- c('x', 'y')
> meuse.grid <- as(meuse.grid, "SpatialPixels")
> image(meuse.grid, col = 'grey')
> title('grid')
```

Spatial Grid Plot

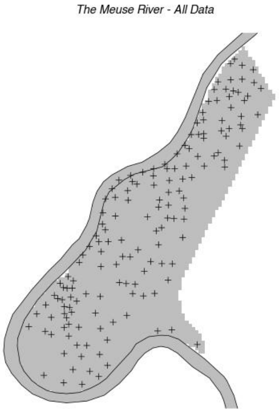


Bringing It all Together

Now, if we pool all of this information, we get a usable visualization of our data set that tells us a lot about the problem we're studying:

```
> image(meuse.grid, col = 'grey')
> plot(meuse.sr, col = 'grey', add = TRUE)
> plot(meuse, add = TRUE)
```

Spatial Pooled Plot



A More Interesting Example

Now let's examine spatial polygon data used to display election results from 2004:

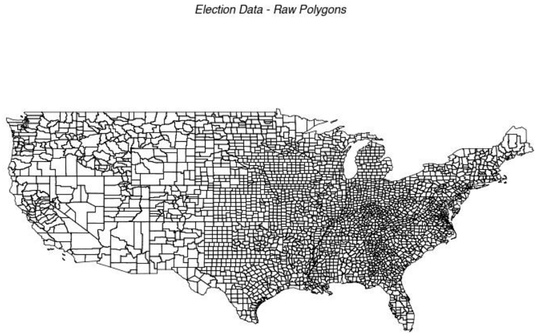
```
> load('Datasets.Rdata')

> class(election)
[1] "SpatialPolygonsDataFrame"
attr(,"package")
[1] "sp"
```

Raw Election Data Polygons

We'll start by just plotting this data set.

```
> plot(election)
```

Raw Election Data Polygons

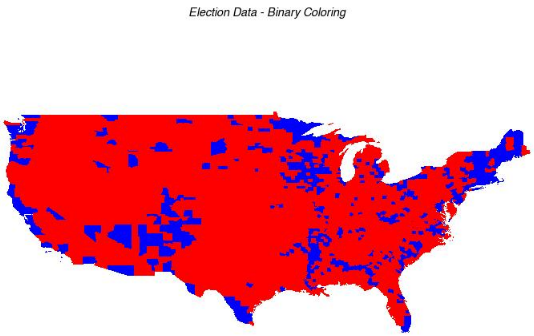


Colored Election Data Polygons

Then we start to add colors reflecting votes:

```
> plot(election,
       col = with(as.data.frame(election),
                  ifelse(Bush > Kerry, 'red', 'blue')),
       border = NA)
```

Colored Election Data Polygons



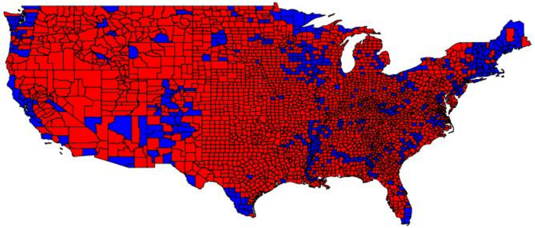
Colored Election Data Polygons

Then add borders:

```
> plot(election,
       col = with(as.data.frame(election),
                  ifelse(Bush > Kerry, 'red', 'blue')),
       border = TRUE)
```

Colored Election Data Polygons

Election Data - Binary Coloring with Borders

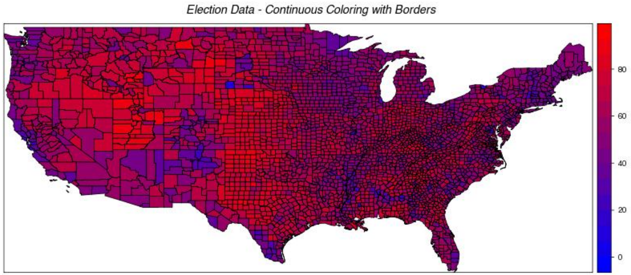


Colored Election Data Polygons

And switch to a continuous scale for Republican percentage votes:

```
> br.palette <- colorRampPalette(c('blue', 'red'),
                                  space = 'rgb')
> spplot(election,
         zcol = 'Bush_pct',
         col.regions = br.palette(100),
         main = 'Election Data')
```

Colored Election Data Polygons



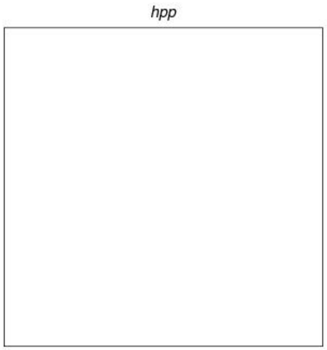
Take Away Message

You can use the data frame I just visualized for any plot of the continental U.S. by simply adding in new columns to the SpatialPolygonsDataFrame.

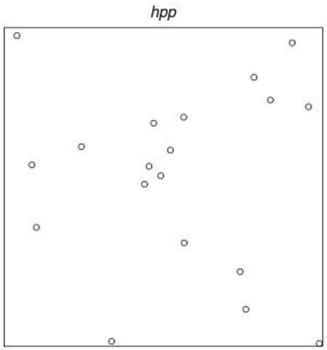
Statistical Models

I want to move away from visualizations and onto statistical methodology. We're going to cover point processes in some depth rather than touch on a lot of topics. Let's just see examples of a point process and then define it formally:

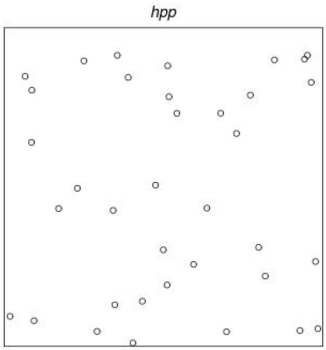
Point Process 1



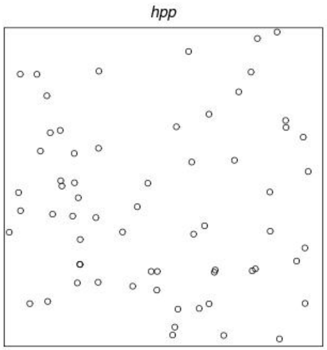
Point Process 2



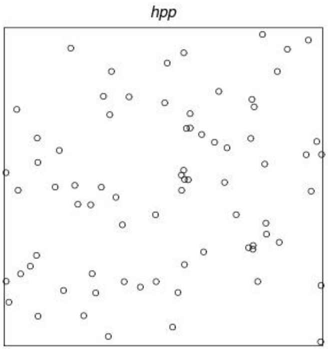
Point Process 3



Point Process 4



Point Process 5



Formal Definition

A point process is a stochastic process that emits points in an N-dimensional space.

Is This Really New?

Couldn't we think of an n-dimensional Gaussian as emitting points in space?

Is This Really New?

Yes, but the Gaussian doesn't have very interesting properties for spatial analyses. Instead of Gaussians and their ilk, we want to understand two things:

- What statistical distributions are relevant to understanding the properties of real data that's distributed in space?

Is This Really New?

Yes, but the Gaussian doesn't have very interesting properties for spatial analyses. Instead of Gaussians and their ilk, we want to understand two things:

- What statistical distributions are relevant to understanding the properties of real data that's distributed in space?
- What statistical tests are relevant to inferring which sort of distribution is behind our empirical data?

Complete Spatial Randomness

The most basic point process we want to understand is one in which points are distributed without rhyme or reason. We call this complete spatial randomness or CSR.

The Homogeneous Poisson Point Process

- Formalization of intuitions about “random” spatial distribution.

The Homogeneous Poisson Point Process

- Formalization of intuitions about “random” spatial distribution.
- Emits λ points over an area A in an N-dimensional space.

The Homogeneous Poisson Point Process

- Formalization of intuitions about "random" spatial distribution.
- Emits λ points over an area A in an N-dimensional space.
- Coordinates in each of these N dimensions are uniformly distributed and independent of other dimensions.

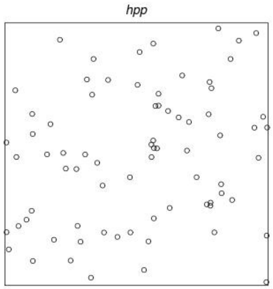
Simulating an HPP

Use rpoispp() to draw data from a Poisson point process. You can specify a constant intensity as a number or a variable intensity as a function of the x and y coordinates.

Simulating an HPP

```
> hpp <- rpoispp(3)
> class(hpp)
[1] "ppp"
> hpp$x
[1] 0.6505647 0.7570645 0.5343773
> hpp$y
[1] 0.3068578 0.9638980 0.9664025
```

Simulating an HPP



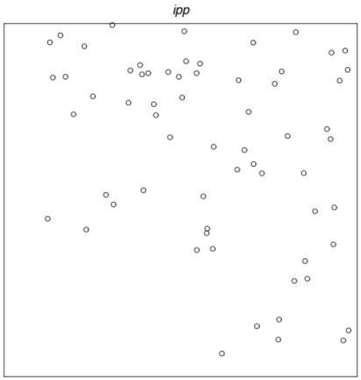
The Inhomogeneous Poisson Process

The inhomogeneous Poisson process is an extension of the HPP where the intensity can vary over space as a function of position.

Simulating an IPP

```
ipp <- rpoispp(function(x,y) {intensity * (x^2 + y^2)})
plot(ipp)
```

Simulating an IPP



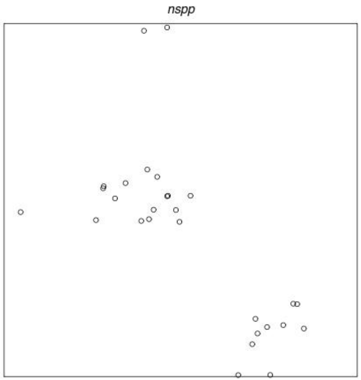
The Neyman-Scott Process

From David Diez: A Neyman-Scott process is basically a homogeneous Poisson process with each parent point replaced with k uniformly distributed children points centered around it.

Simulating an NSPP

```
> nspp <- rNeymanScott(kappa = 10,
                       rmax = 0.1,
                       function(x,y)
                       {
                         runifdisc(5, 0.1,
                                   centre = c(x, y))
                       })
> plot(nspp)
```

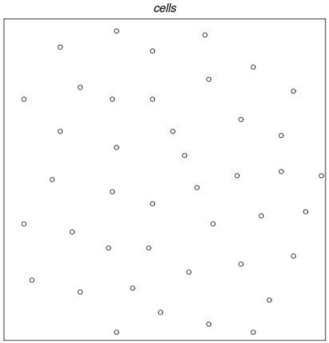
Simulating an NSPP



Some Real Spatial Point Process Datasets

```
> data(cells)
> plot(cells)
```

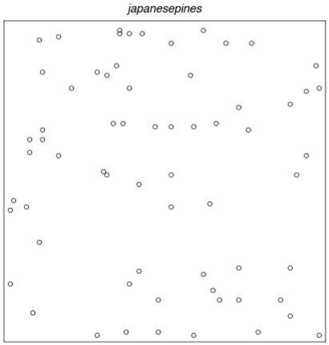
The Cells Dataset



Some Real Spatial Points Datasets

```
> data(japanesepines)
> plot(japanesepines)
```

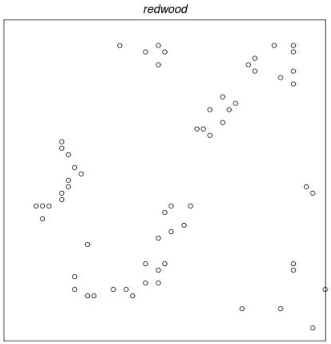
The Japanese Pines Dataset



Some Real Spatial Points Datasets

```
> data(redwood)
> plot(redwood)
```

The Redwood Dataset



Inference and Hypothesis Testing

We've discussed some data generating processes. How can we test whether these processes could plausibly have generated empirical data? We need novel test statistics.

Inference and Hypothesis Testing

We might ask of our data: what percentage of points have their nearest neighbor at a distance r? Under complete spatial randomness, this has a well-defined value, so we can compare empirical results with theoretical expectations as a test statistic.

$G(r)$: A Definition

We define $G(r)$ as follows:

- Let $d_i = min_j\{d_{i,j}, \forall j \neq i\}$.

$G(r)$: A Definition

We define $G(r)$ as follows:

- Let $d_i = min_j\{d_{i,j}, \forall j \neq i\}$.
- Then
$$\hat{G}(r) = \frac{\#\{d_i : d_i \leq r, \forall i\}}{n}.$$

$G(r)$: Theoretical Expected Value

Under CSR (i.e. an HPP with intensity λ), the expected value is:

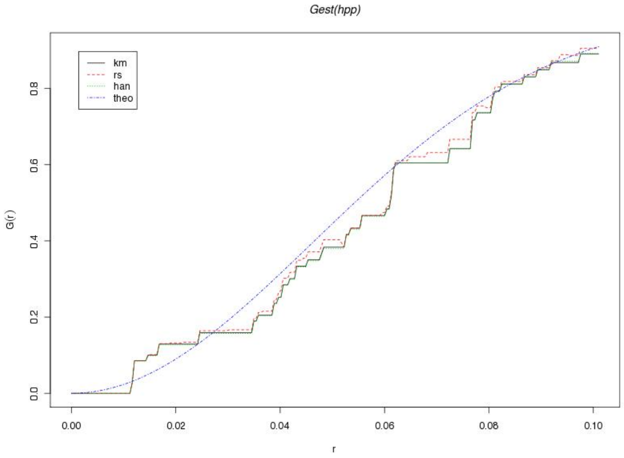
$$G(r) = 1 - e^{-\lambda \pi r^2}$$

Testing $G(r)$ in R: Gest()

With this in hand, we can compare this theoretical to an empirical estimator and see if there are systematic deviations from the expected values using the function Gest():

```
> plot(Gest(hpp))
```

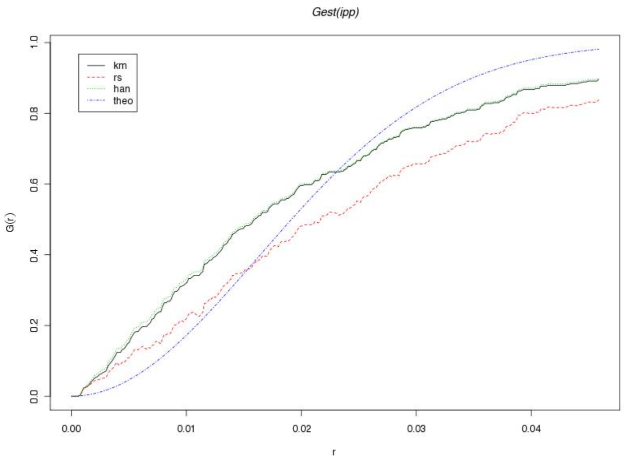
Testing $G(r)$ on a HPP



Testing $G(r)$ on an IPP

```
> plot(Gest(ipp))
```

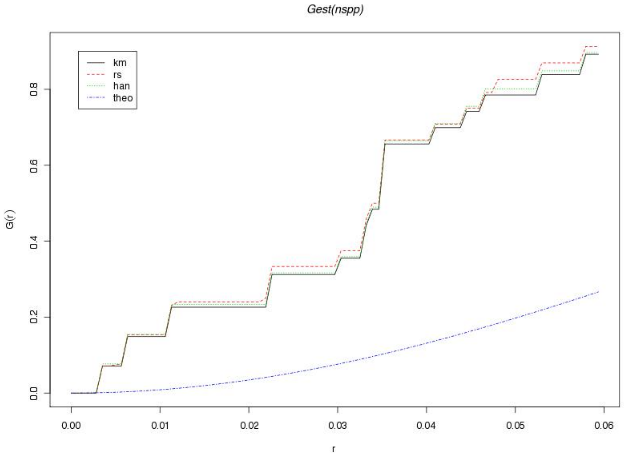
Testing $G(r)$ on an IPP



Testing $G(r)$ on a NSPP

```
> plot(Gest(nspp))
```

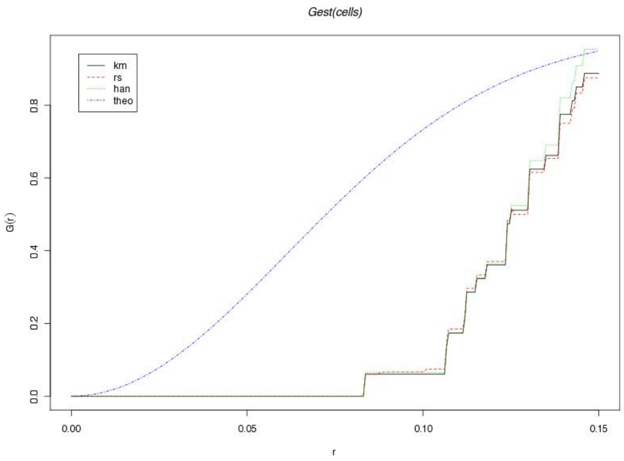
Testing $G(r)$ on a NSPP



Testing $G(r)$ on the Cells Data

```
> plot(Gest(cells))
```

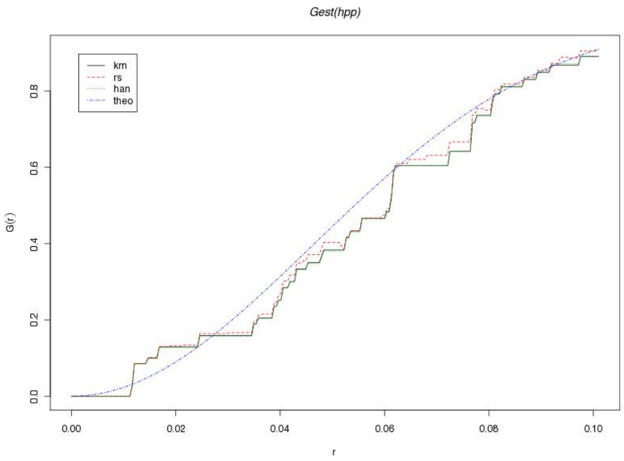
Testing $G(r)$ on the Cells Data



Testing $G(r)$ on the Japanese Pines Data

```
> plot(Gest(japanesepines))
```

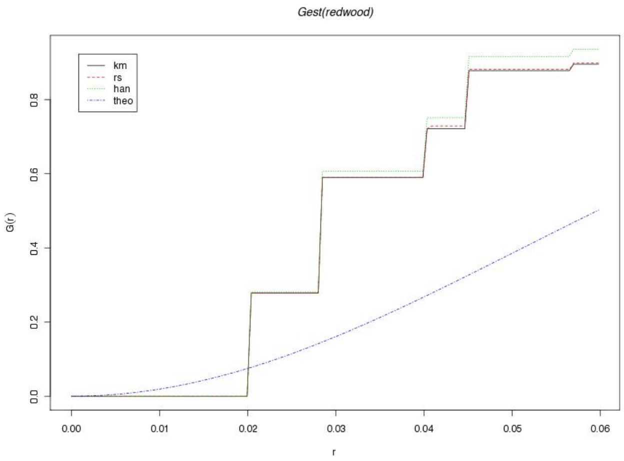
Testing $G(r)$ on the Japanese Pines Data



Testing $G(r)$ on the Redwood Data

```
> plot(Gest(redwood))
```

Testing $G(r)$ on the Redwood Data



$F(r)$: A Definition

We define $F(r)$ as follows. Given any point, how far away is the nearest emitted point? This is the empty space measure. Under CSR, the expected value is:

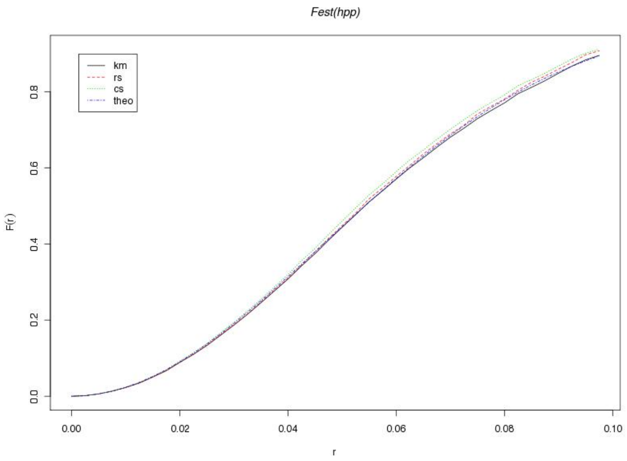
$$F(r) = 1 - e^{-\lambda \pi r^2}$$

Testing $F(r)$ in R

With this in hand, we can compare this theoretical to an empirical estimator and see if there are systematic deviations from the expected values using Fest():

```
> plot(Fest(hpp))
```

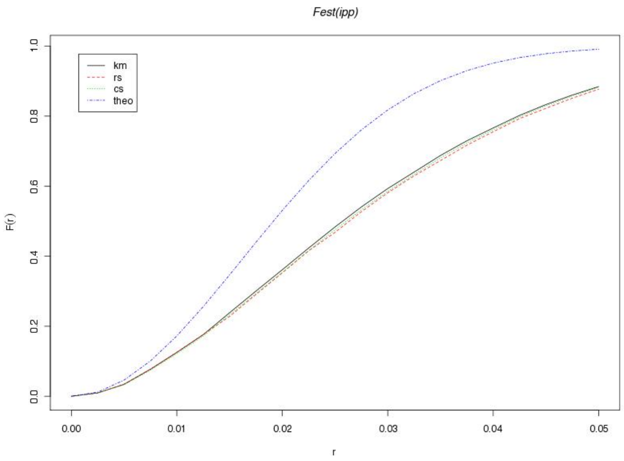
Testing $F(r)$ on an HPP



Testing $F(r)$ on an IPP

```
> plot(Fest(ipp))
```

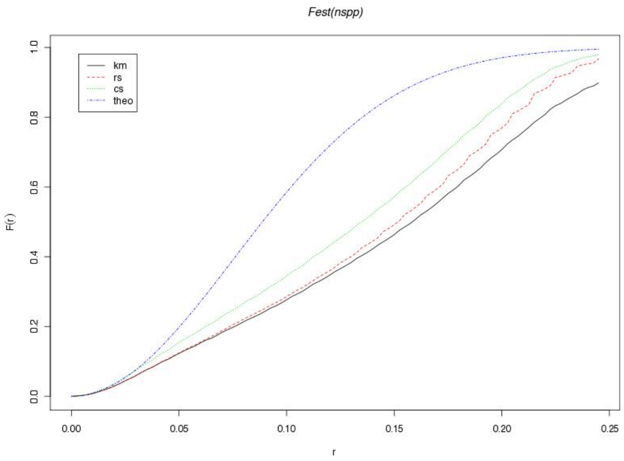
Testing $F(r)$ on an IPP



Testing $F(r)$ on a NSPP

```
> plot(Fest(nspp))
```

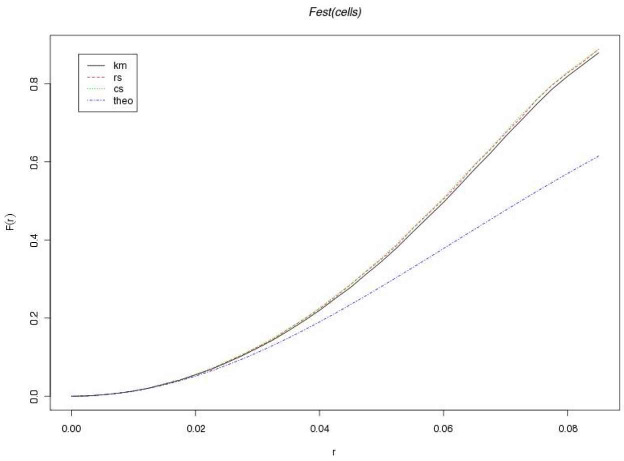
Testing $F(r)$ on a NSPP



Testing $F(r)$ on the Cells Data

```
> plot(Fest(cells))
```

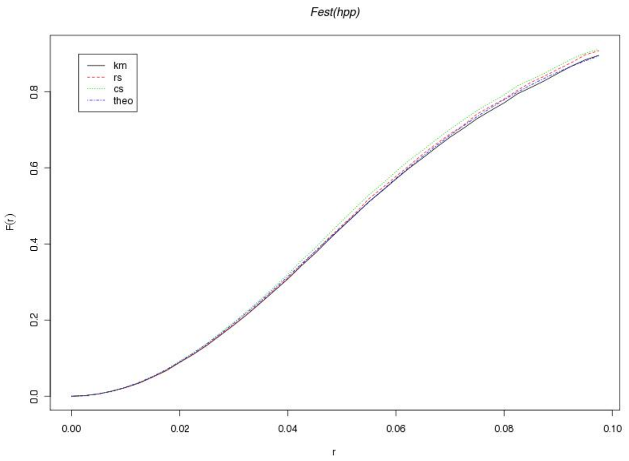
Testing $F(r)$ on the Cells Data



Testing $F(r)$ on the Japanese Pines Data

```
> plot(Fest(japanesepines))
```

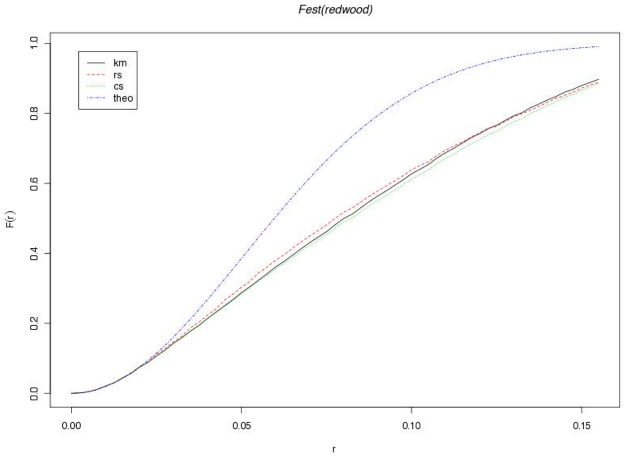
Testing $F(r)$ on the Japanese Pines Data



Testing $F(r)$ on the Redwood Data

```
> plot(Fest(redwood))
```

Testing $F(r)$ on the Redwood Data



$K(r)$: A Definition

Here's one more function for hypothesis tests:

- $N(s) =$ Number of Points within a Distance s.

$K(r)$: A Definition

Here's one more function for hypothesis tests:

- $N(s)$ = Number of Points within a Distance s.
- $K(r) = \lambda^{-1}\mathbb{E}[N(s)]$.

$K(r)$: Theoretical Expected Value

Under CSR (i.e. an HPP with intensity λ), the expected value is:

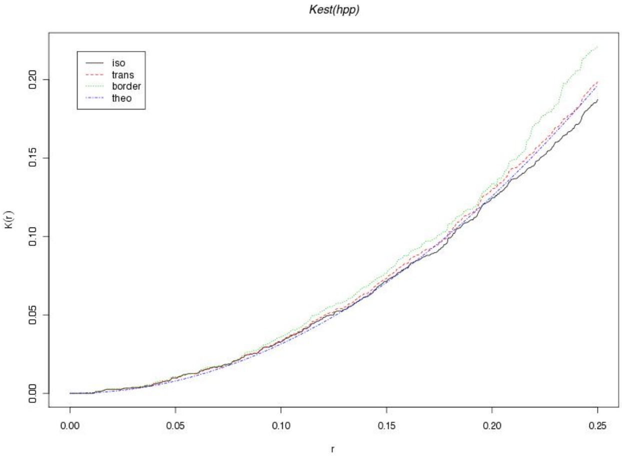
$$K(r) = \pi r^2$$

Testing $K(r)$ in R

With this in hand, we can compare this theoretical to an empirical estimator and see if there are systematic deviations from the expected values using Kest():

```
> plot(Kest(hpp))
```

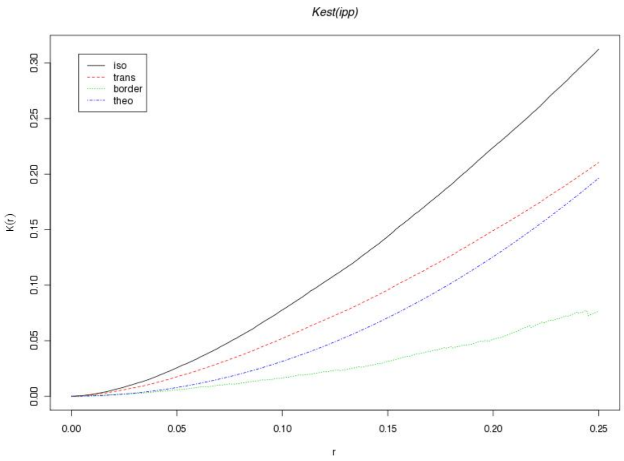
Testing $K(r)$ on a HPP



Testing $K(r)$ on an IPP

```
> plot(Kest(ipp))
```

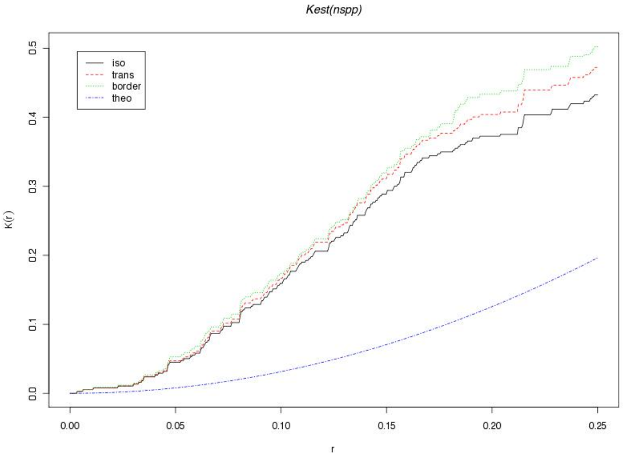
Testing $K(r)$ on an IPP



Testing $K(r)$ on a NSPP

```
> plot(Kest(nspp))
```

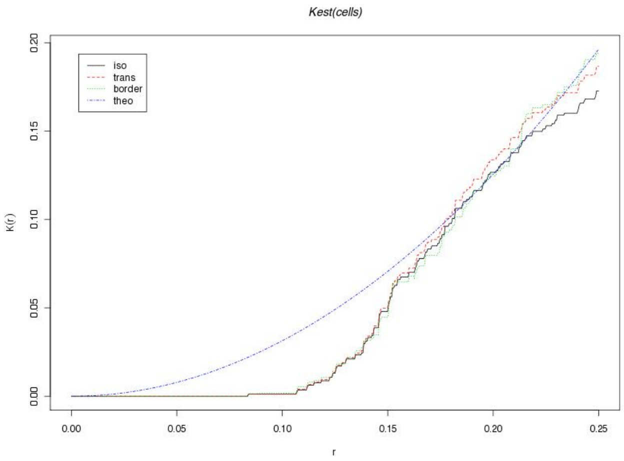
Testing $K(r)$ on a NSPP



Testing $K(r)$ on the Cells Data

```
> plot(Kest(cells))
```

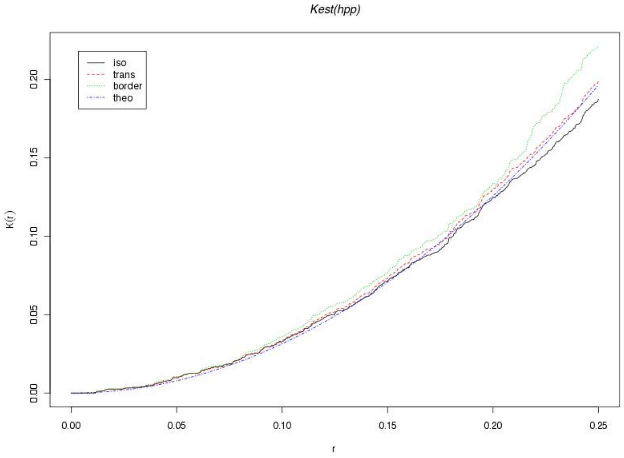
Testing $K(r)$ on the Cells Data



Testing $K(r)$ on the Japanese Pines Data

```
> plot(Kest(japanesepines))
```

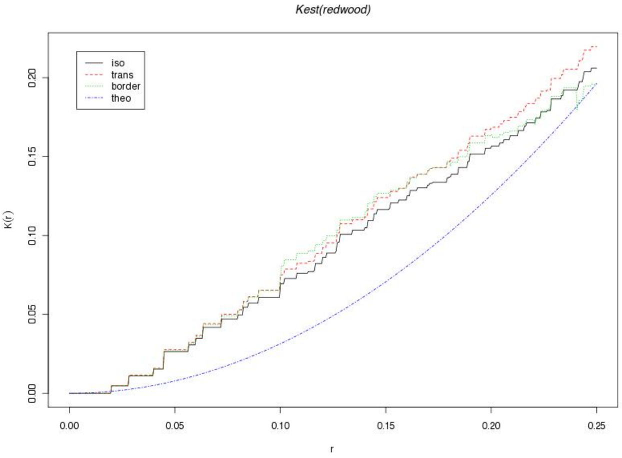
Testing $K(r)$ on the Japanese Pines Data



Testing $K(r)$ on the Redwood Data

```
> plot(Kest(redwood))
```

Testing $K(r)$ on the Redwood Data



Take Away Message

You can easily run hypothesis tests to assess the plausibility that standard point process models generated your empirical data. You can see if CSR holds or whether other canonical structures (e.g. attraction and repulsion) exist.

Statistical Models for Non-IID Data

Finally, there's another topic I can only just touch upon: how do we cope with data whose location is not of interest, but where the points have measurements whose distribution depends upon location?

Statistical Models for Non-IID Data

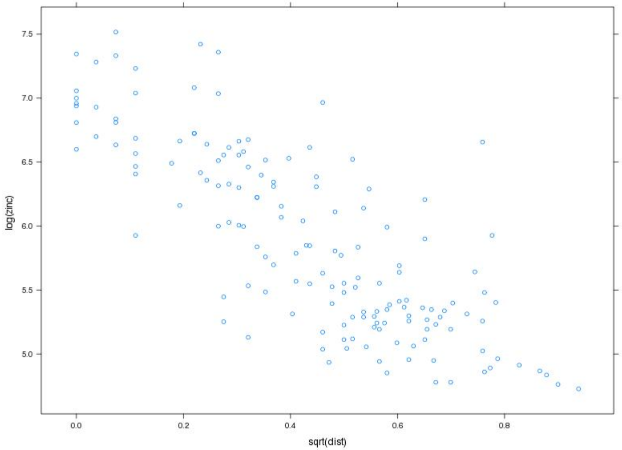
This is fraught with difficulty, because the data set likely fails the IID assumption, and so the true number of observations is smaller than you might think. This will not usually change the maximum likelihood estimators, but will substantially change the variance of our inferences.

Returning to the Meuse Data

Let's look at the Meuse data again. We'll see if zinc concentration is predicted by a point's distance from the river bank:

```
> xyplot(log(zinc) ~ sqrt(dist),
         as.data.frame(meuse))
```

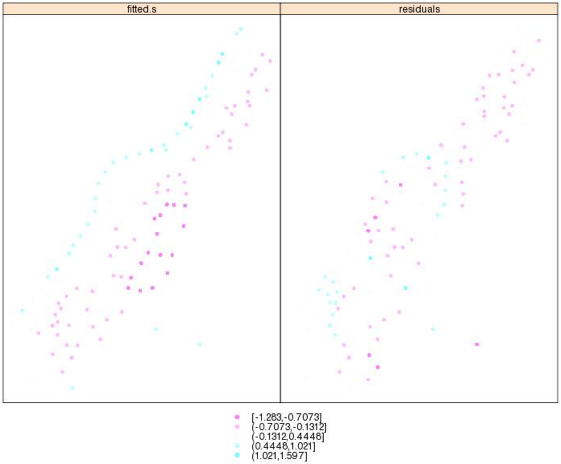
Distance Predicts Zinc Concentration



Build a Linear Model

```
> zn.lm <- lm(log(zinc) ~ sqrt(dist),
         data = as.data.frame(meuse))
> meuse$fitted.s <- predict(zn.lm, meuse)
         - mean(predict(zn.lm, meuse))
> meuse$residuals <- residuals(zn.lm)
> spplot(meuse, c('fitted.s', 'residuals'))
```

Problems with the Linear Model Predictions



Problems with the Linear Model Predictions

Notice the correlated errors for nearby points: the error terms for our model are not correct when they do not take into account the correlations between nearby points.

Variograms: A Definition

We can formalize this concern using variograms. The variogram is a measure of the changes in the variance of our measurement as a function of the distance between points. To create variograms, we define the semivariance function for a given distance *h*:

$$\gamma(h) = \frac{1}{2}\mathbb{E}[(Z(s) - Z(s+h))^2]$$

Variograms

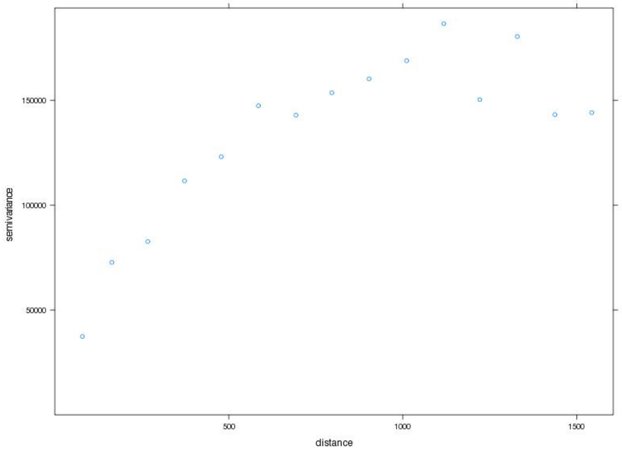
Imagine that our data has no spatial structure: for example, suppose that the value at any point s is $Z(s) = c + \epsilon$, where epsilon is Gaussian noise. Under this assumption, the variance between points is constant and the variogram is a constant corresponding to the variance of ϵ.

Empirical Variograms

We can plot variograms for empirical data to see whether this constant variance across space holds:

```
> coordinates(meuse) <- c('x', 'y')
> plot(variogram(zinc ~ 1, meuse))
```

Empirical Variograms



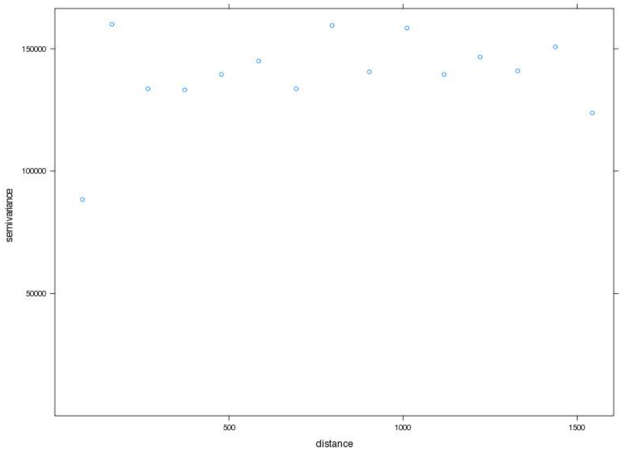
Permuted Variogram

By itself, this plot suggests that the variabiity is not constant over space, but we'd like a better sense of how this compares with arbitrary, but similar, data. We can permute the spatial labels for our zinc measurements and then plot a new variogram to do this:

Permuted Variograms

```
> meuse$permuted.zinc <- sample(meuse$zinc, nrow(meuse))
> plot(variogram(permuted.zinc ~ 1, meuse))
```

Permuted Variogram



Permuted Variogram

You can imagine running a large sample of permutations to turn this into a hypothesis test for spatial correlation acting as a hidden variable.

Exploiting Spatial Correlation

Clearly spatial correlation can be a problem. But can we use spatial correlation to improve predictions, rather than weaken inference?

Inverse Distance Weighted Interpolation

A naive method is to perform a sort of k-nearest neighbors in which we up-weight near points and down-weight far points as part of a routine for interpolation concerning unobserved locations.

Inverse Distance Weighted Interpolation

Formally, we use the following interpolation function:

$$\hat{Z}(s_0) = \frac{\sum_{i=1}^{n} w(s_i) Z(s_i)}{\sum_{i=1}^{n} w(s_i)},$$

where

$$w(s_i) = ||s_i - s_0||^{-p}$$

for some (possibly fitted) p.

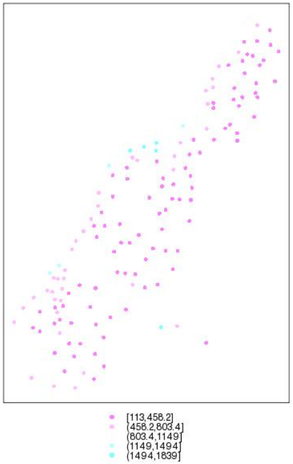
Inverse Distance Weighted Interpolation in R

```
> meuse.grid <- as(meuse.grid, 'SpatialPixelsDataFrame')

> idw.out <- idw(zinc ~ 1, meuse, meuse.grid, idp = 2.5)
```

Refresh Your Memory of The Empirical Data

```
> spplot(meuse, z = 'zinc')
```

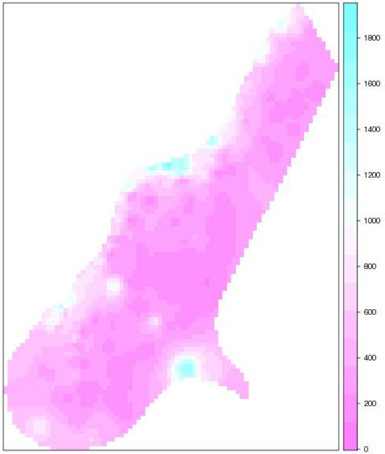
The Empirical Data



IDW Predictions

```
spplot(idw.out, z = 'var1.pred')
```

IDW Predictions



Other Topics

Other items worth looking into:

- Moran's I and Geary's C

Other Topics

Other items worth looking into:

- Moran's I and Geary's C
- Kriging

References

- Yuri M. Zhukov: Slides

References

- Yuri M. Zhukov: Slides
- David Diez: Slides

References

- Yuri M. Zhukov: Slides
- David Diez: Slides
- Applied Spatial Data Analysis with R: Book