Spatial analysis lab

see also

Constructing weight matrices

```
library(spdep)
```

```
nyfile <- system.file("etc/misc/nydata.dbf", package = "spdep")
nydata0 <- read.dbf(nyfile)  ## read.dbf is from the foreign package, auto-loaded by spdep
head(nydata0)
```

```
##          AREANAME     AREAKEY     X      Y POP8 TRACTCAS  PROPCAS
## 1 Binghamton city 36007000100 4.069 -67.35 3540     3.08 0.000870
## 2 Binghamton city 36007000200 4.639 -66.86 3560     4.08 0.001146
## 3 Binghamton city 36007000300 5.709 -66.98 3739     1.09 0.000292
## 4 Binghamton city 36007000400 7.614 -66.00 2784     1.07 0.000384
## 5 Binghamton city 36007000500 7.316 -67.32 2571     3.06 0.001190
## 6 Binghamton city 36007000600 8.559 -66.93 2729     1.06 0.000388
##   PCTOWNHOME PCTAGE65P       Z AVGIDIST PEXPOSURE
## 1     0.3277    0.1466  0.1420   0.2374     3.167
## 2     0.4268    0.2351  0.3555   0.2087     3.039
## 3     0.3377    0.1380 -0.5817   0.1709     2.838
## 4     0.4616    0.1189 -0.2963   0.1406     2.643
## 5     0.1924    0.1416  0.4569   0.1578     2.759
## 6     0.3652    0.1411 -0.2812   0.1726     2.848
```

```
nydata <- nydata0  ## make a copy to turn into a 'sp' object
coordinates(nydata) <- c("X", "Y")  ## set coordinates
nycoord <- coordinates(nydata)  ## retrieve coordinates
## or: nycoord <- nydata[,c('X','Y')] *before* setting coordinates(nydata)
```

- Use `View()`/`plot()`/`summary()`/etc. to investigate `nydata`. (Use `?nydata`. Or try

```
library(ggplot2)
theme_set(theme_bw())
ggplot(nydata0, aes(x = X, y = Y, size = PROPCAS, color = PEXPOSURE)) + geom_point(alpha = (
```

The "neighbour-list" (`nb`) and "weight list" (`listw`) structures are the basic components of SAR/CAR/Moran's I/etc.

Explore the available functions for converting to and from these object types, and doing things with them:

```
## use regular expressions! ^='beginning of line', $='end of line'
apropos("(^nb2|2nb$)")
```

```
##  [1] "cell2nb"      "graph2nb"     "gridIndex2nb" "knn2nb"
##  [5] "nb2blocknb"   "nb2INLA"      "nb2lines"     "nb2listw"
##  [9] "nb2mat"       "nb2WB"        "poly2nb"      "read.gwt2nb"
## [13] "tri2nb"
```

```
apropos("(^listw2|2listw$)")
```

```
##  [1] "listw2lines"    "listw2mat"      "listw2sn"       "listw2star"
##  [5] "listw2U"        "listw2WB"       "mat2listw"      "nb2listw"
##  [9] "read.dat2listw" "sn2listw"
```

```
methods(class = "nb")
```

```
## [1] aggregate.nb edit.nb      plot.nb      print.nb     subset.nb
## [6] summary.nb
```

```
methods(class = "listw")
```

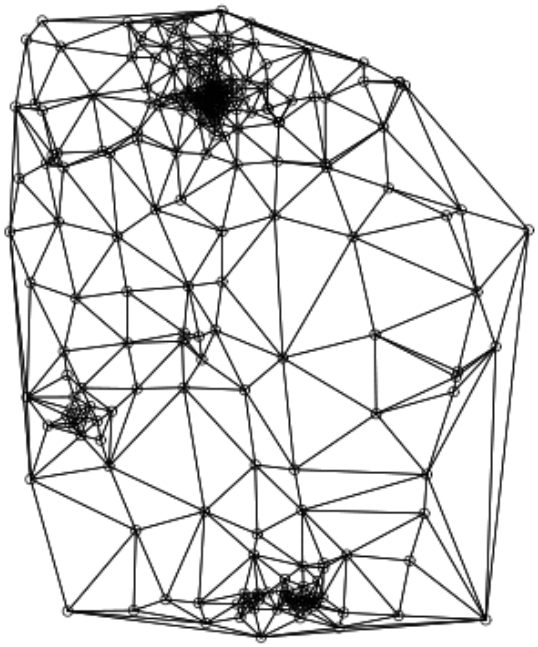
```
## [1] lag.listw     plot.listw    print.listw   subset.listw  summary.listw
```

Construct Delaunay tesselation (triangulation)/Voronoi diagram: graph of nearest neighbors.

```
library(deldir)
nynb <- tri2nb(nycoord)
```

```
##
##      PLEASE NOTE:  The components "delsgs" and "summary" of the
##      object returned by deldir() are now DATA FRAMES rather than
##      matrices (as they were prior to release 0.0-18).
##      See help("deldir").
##
##      PLEASE NOTE: The process that deldir() uses for determining
##      duplicated points has changed from that used in version
##      0.0-9 of this package (and previously). See help("deldir").
```

```
plot(nynb, nycoord)  ## SLOW
```

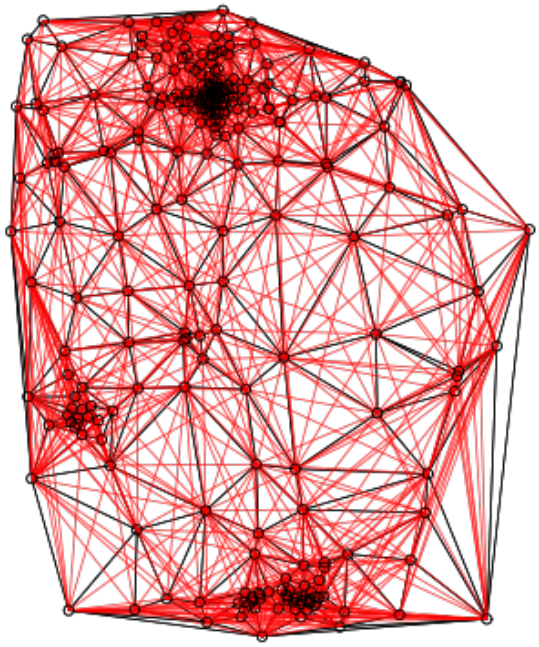


```
## compute 1st- and 2d-order neighbours:
(col.lags <- nblag(nynb, 2))
```

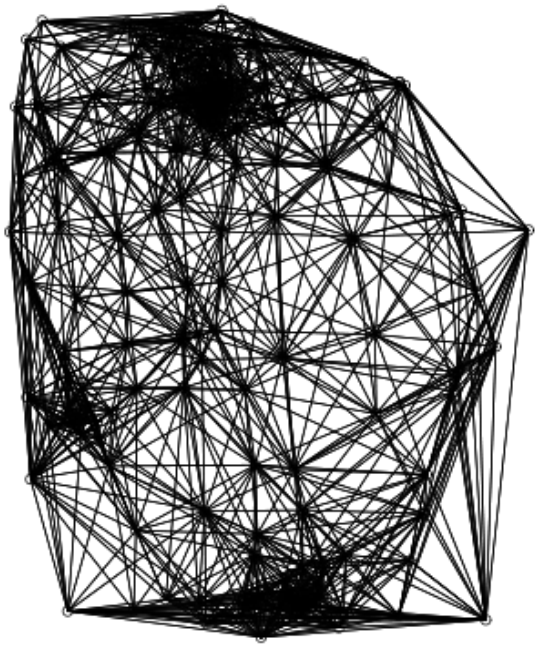
```
## [[1]]
## Neighbour list object:
## Number of regions: 281
## Number of nonzero links: 1654
## Percentage nonzero weights: 2.095
## Average number of links: 5.886
##
## [[2]]
## Neighbour list object:
## Number of regions: 281
## Number of nonzero links: 3564
## Percentage nonzero weights: 4.514
```

```
## Average number of links: 12.68
##
## attr(,"call")
## nblag(neighbours = nynb, maxlag = 2)
```

```
plot(nynb, nycoord)
plot(col.lags[[2]], nycoord, add = TRUE, col = adjustcolor("red", alpha = 0.5))
```

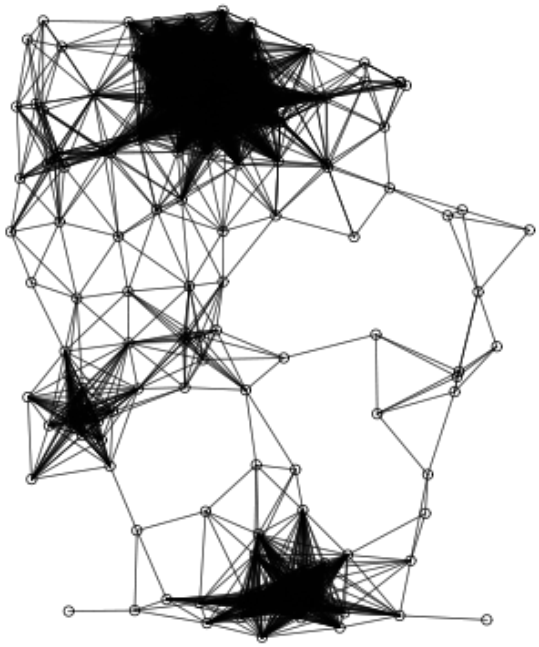


```
cuml <- nblag_cumul(col.lags)  ## collapse first- & second-order neighbors
plot(cuml, nycoord)
```



Distance-based weight matrix:

```
nymat2 <- as.matrix(dist(nycoord)) < 20
nymat2[] <- as.numeric(nymat2)  ## convert without losing matrix structure
## note 'dist' takes alternative metrics such as 'manhattan'
listw_NY <- mat2listw(nymat2)
plot(listw_NY, nycoord, col = adjustcolor("black", alpha = 0.5))
```



Use pre-computed adjacency matrix:

```
nyadjfile <- system.file("etc/misc/nyadjwts.dbf", package = "spdep")
nyadjdat <- read.dbf(nyadjfile)
```

Lots of messages, generated by trying to make fields unique

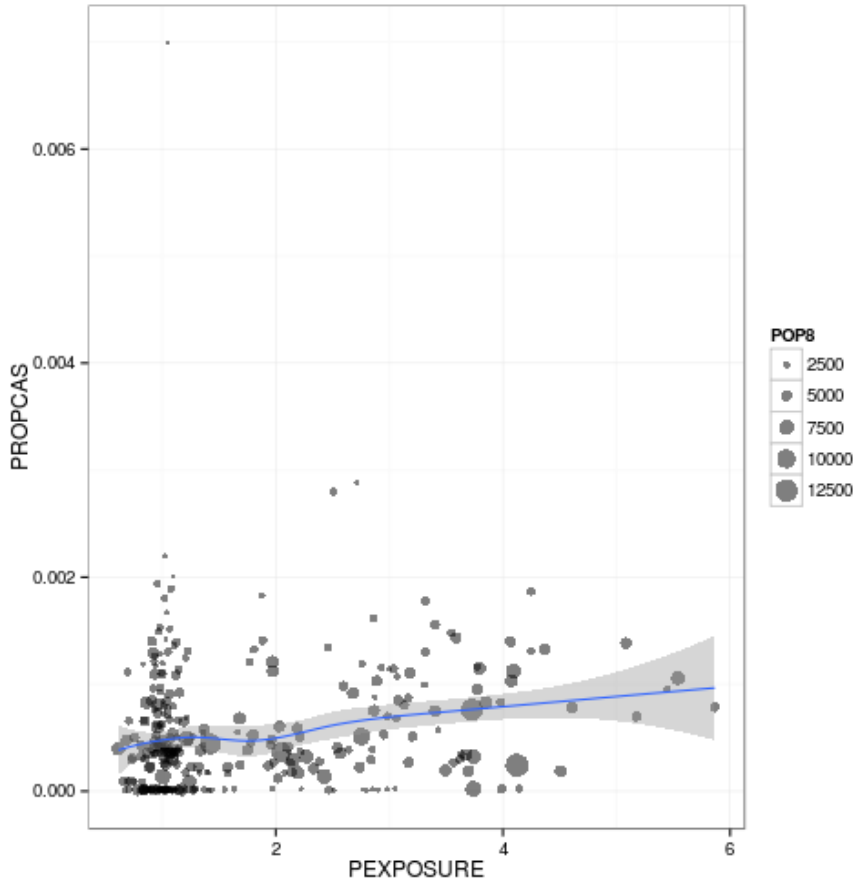
```
nyadjmat <- as.matrix(nyadjdat[, -1])  ## first column is an ID variable
ID <- names(nyadjdat)[-1]
## check that area keys and IDs are the same ...
identical(substring(ID, 2, 10), substring(as.character(nydata$AREAKEY), 2, 10))
```

```
## [1] TRUE
```

```
nyadjlw <- mat2listw(nyadjmat, ID)
listw_NY <- nb2listw(nyadjlw$neighbours, style = "B")
```

```
## plot prop. cases vs exposure, weight smooth and adjust point size by
## pop size
ggplot(nydata0, aes(x = PEXPOSURE, y = PROPCAS)) + geom_point(aes(size = POP8),
    alpha = 0.5) + geom_smooth(aes(weight = POP8, method = "loess"))

## geom_smooth: method="auto" and size of largest group is <1000, so using
## loess. Use 'method = x' to change the smoothing method.
```




```
lm1 <- lm(PROPCAS ~ PEXPOSURE, weights = POP8, data = nydata0)
summary(lm1)

##
## Call:
## lm(formula = PROPCAS ~ PEXPOSURE, data = nydata0, weights = POP8)
##
```

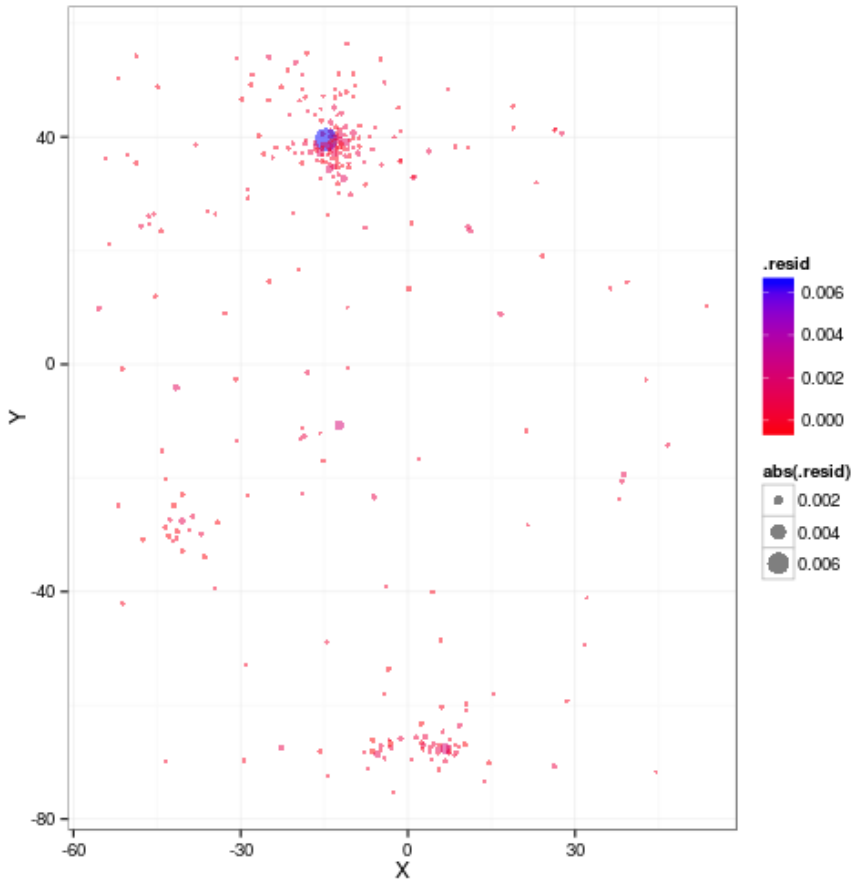
```
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.06503 -0.02029 -0.00579  0.01748  0.11774
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.62e-04   5.23e-05    6.92  3.1e-11 ***
## PEXPOSURE   1.02e-04   2.29e-05    4.46  1.2e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0284 on 279 degrees of freedom
## Multiple R-squared:  0.0667, Adjusted R-squared:  0.0633
## F-statistic: 19.9 on 1 and 279 DF,  p-value: 1.17e-05
```

```
lm.morantest(lm1, listw_NY)  ## reject null (just ...)
```

```
##
##  Global Moran's I for regression residuals
##
## data:
## model: lm(formula = PROPCAS ~ PEXPOSURE, data = nydata0, weights =
## POP8)
## weights: listw_NY
##
## Moran I statistic standard deviate = 1.685, p-value = 0.046
## alternative hypothesis: greater
## sample estimates:
## Observed Moran's I        Expectation           Variance
##           0.053705          -0.006257           0.001266
```

See also `lm.LMtests` and `moran.test`

```
lm1F <- data.frame(fortify(lm1), subset(nydata0, select = c(X, Y)))
ggplot(lm1F, aes(x = X, y = Y)) + geom_point(aes(size = abs(.resid), colour = .resid),
    alpha = 0.5) + scale_colour_gradient(low = "red", high = "blue")
```

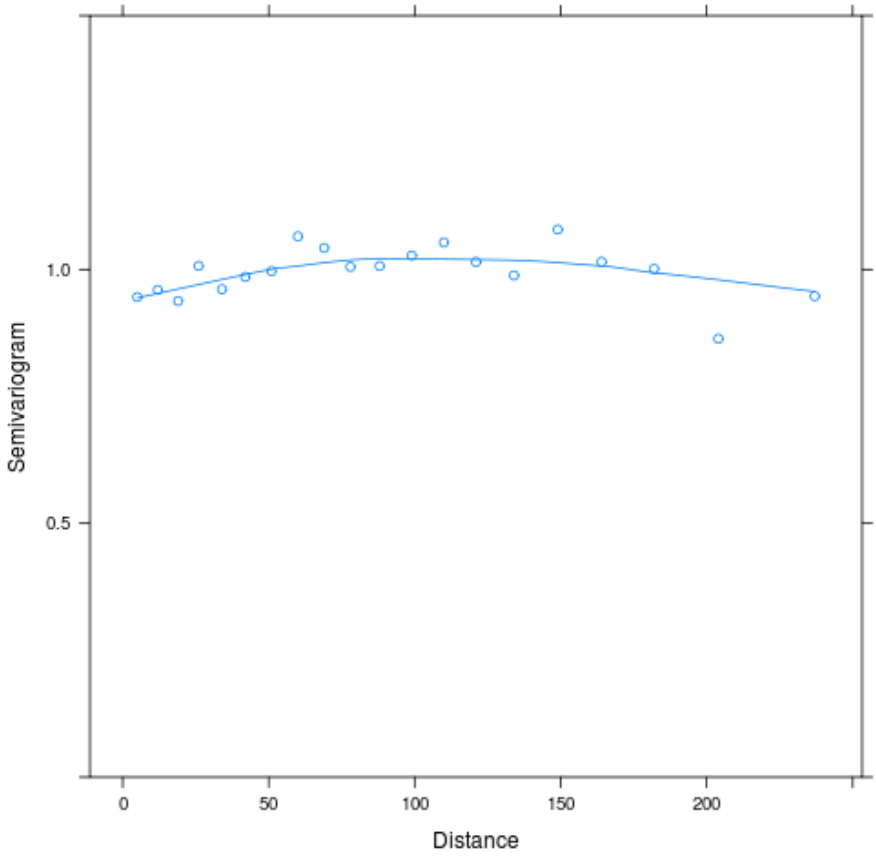


We'd probably better check out that giant residual and do something about it

...

Geostatistical

```
library(nlme)
g1 <- gls(PROPCAS ~ PEXPOSURE, weights = varFixed(~1/POP8), data = nydata0)
plot(Variogram(g1), ylim = c(0, 1.5))
```

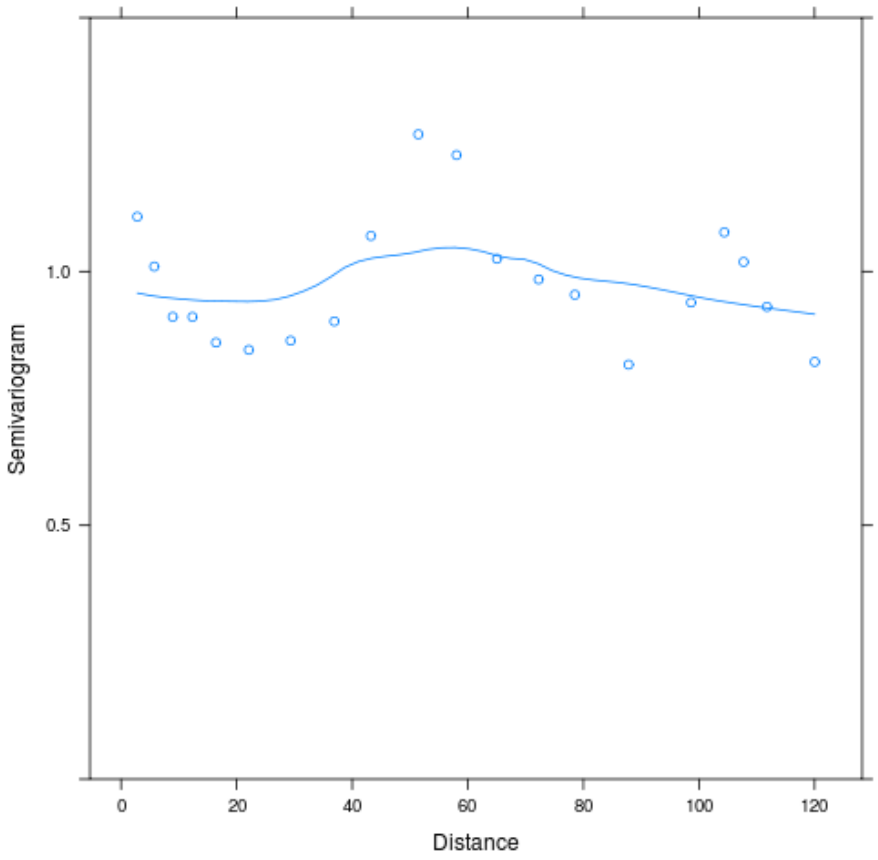


There doesn't actually seem to be much going on here ...

```
g1M <- update(g1, method = "ML")
g2 <- update(g1, . ~ . + poly(X, Y, degree = 2))
g2M <- update(g2, method = "ML")
anova(g1M, g2M, test = TRUE)
```

```
##     Model df   AIC   BIC logLik   Test L.Ratio p-value
## g1M     1  3 -3463 -3453   1735
## g2M     2  8 -3465 -3436   1740 1 vs 2   11.38  0.0443
```

```
plot(Variogram(g2, form = ~X + Y), ylim = c(0, 1.5))
```



. . . and there's even less when we subtract the spatial trend.

Nevertheless we will forge ahead and try to fit a spatial model.

If you have time, try out the `likfit` function from the `geoR` package . . .