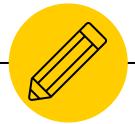
Exploratory Spatial Data Analysis in the tidyverse



Exploratory Data Analysis

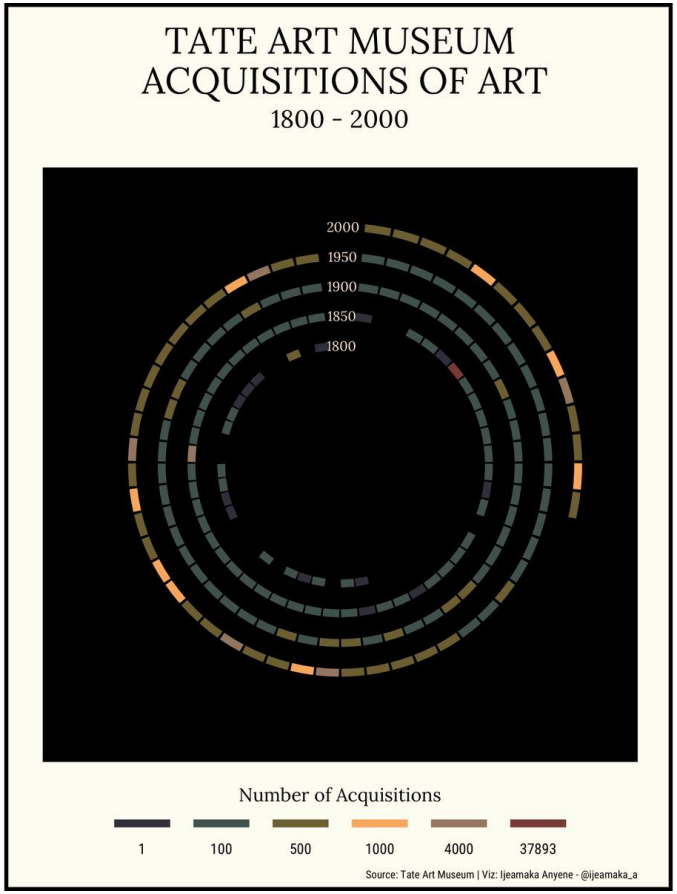
"Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to:

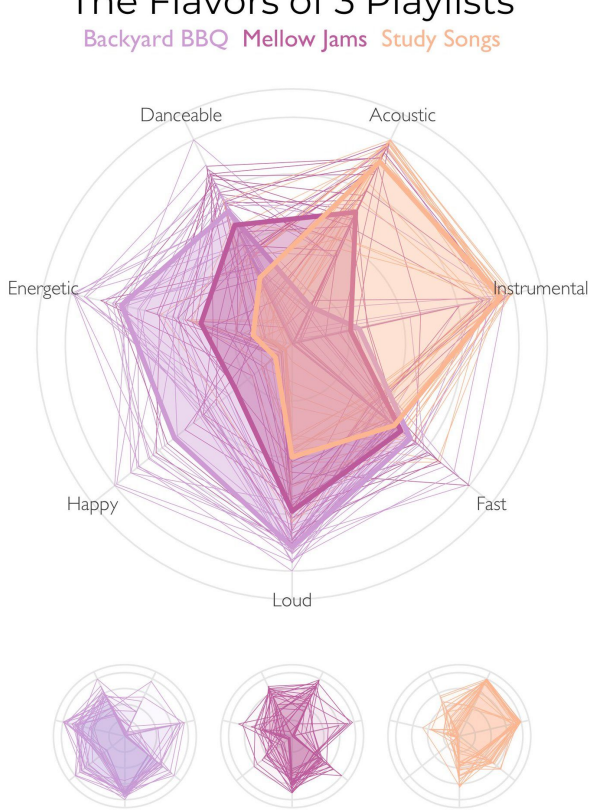
1. *maximize insight into a data set;*
2. *uncover underlying structure;*
3. *extract important variables;*
4. *detect outliers and anomalies…"*

– NIST *Handbook*

Exploratory Data Analysis

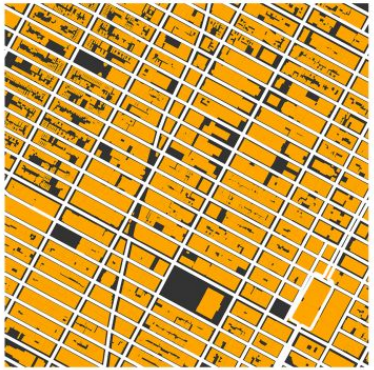
- “Discover potentially explicable patterns”
  - (Good, 1983)
- Emphasis on data visualization
- Use of descriptive statistics
- Discovering outliers


TATE ART MUSEUM
ACQUISITIONS OF ART
1800 - 2000
2000
1950
1900
1850
1800
Number of Acquisitions
1
100
500
1000
4000
37893
Source: Tate Art Museum | Viz: Ijeamaka Anyene - @ijeamaka_a

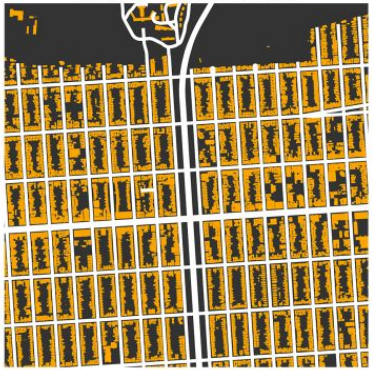

The Flavors of 3 Playlists
Backyard BBQ
Mellow Jams
Study Songs
Danceable
Acoustic
Energetic
Instrumental
Happy
Fast
Loud
Data from Spotify & SpotifyR | Visualization by @Jake_Lawlor1

Exploratory Spatial Data Analysis

NEW YORK

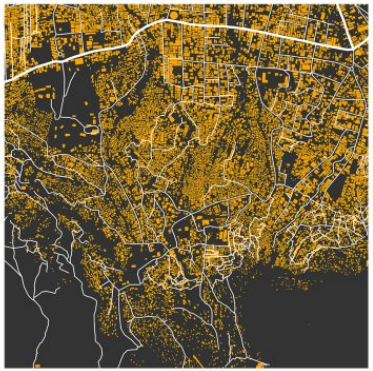


SAN FRANCISCO



MONROVIA

PORT-AU-PRINCE



“Everything is related to everything else,
but near things are more related
than distant things.”

(*Waldo R. Tobler*, 1970)

Exploratory Spatial Data Analysis (ESDA)

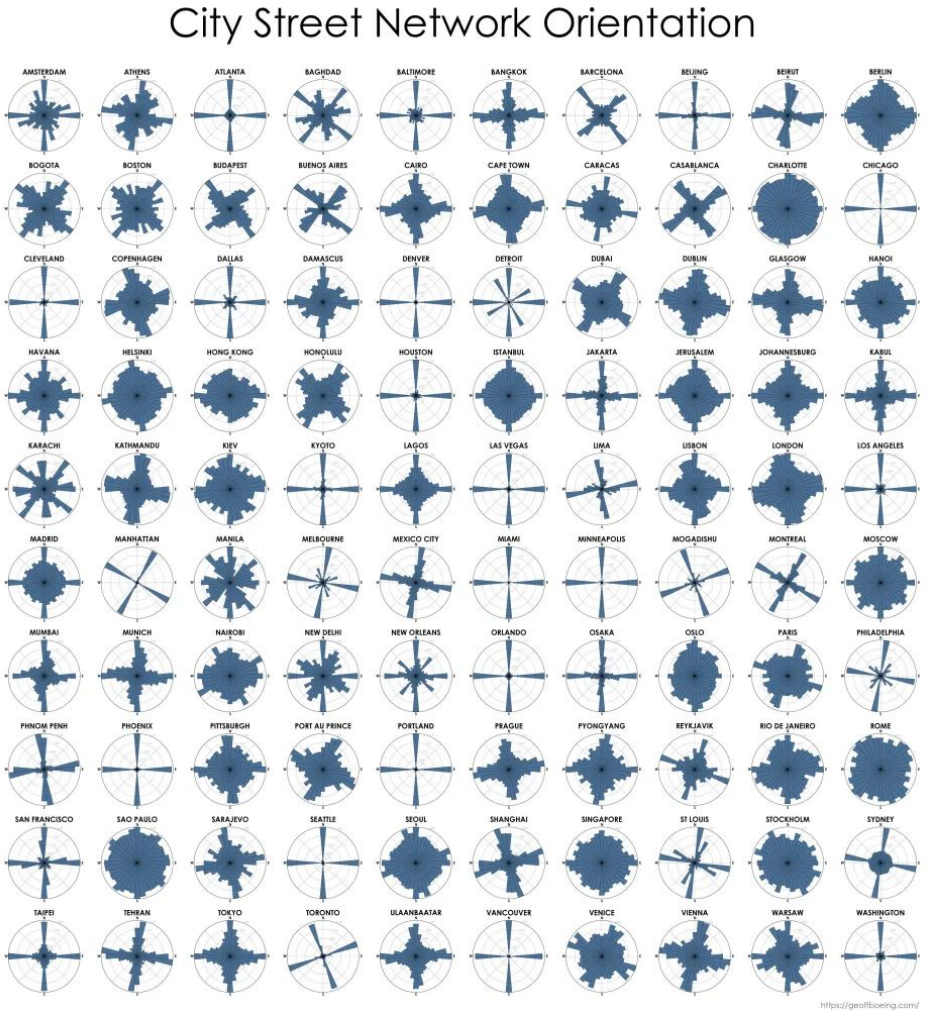
- Extends EDA to spatial relationships
- Asks:
  - Are things randomly distributed?
  - Are there spatial outliers?
  - Are close things more similar?

ESDA

- *EDA* compares a part to the whole
- *ESDA* compares a part to its neighboring parts
- In *ESDA* we evaluate a location to its **neighbors**

How I learned

- Advanced Spatial Analytics w/ Geoff Boeing
- Used Python:
  - geopandas
  - shapely
  - pysal



python, tho...

I want to use #rstats

{spdep}

Spatial Dependence: Weighting Schemes and Statistics

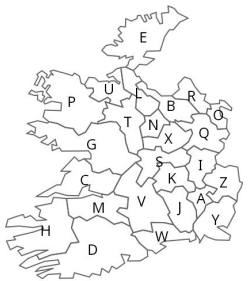
spdep

- Released in 2002
- Designed for {sp}
- Now supports {sf}
- Covers most ESDA stats, weights, and neighbors

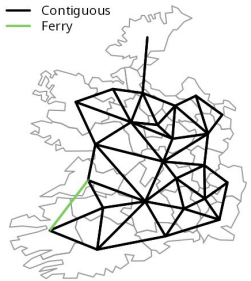
```
xx <- diffnb(nb, lw_unstand$neighbours, verbose=TRUE)
```

```
## Neighbour difference for region id: Clare in relation to id: Kerry
## Neighbour difference for region id: Kerry in relation to id: Clare
```

```
plot(eire_ge1, border="grey60")
plot(nb, coordinates(eire_ge1), add=TRUE, pch=".", lwd=2)
plot(xx, coordinates(eire_ge1), add=TRUE, pch=".", lwd=2, col=3)
```



25 Irish counties



Contiguities

Me & spdep

- In 2002 I was 6
- I “grew up” with the tidyverse
- A perceived paradigm gap

Me & spdep

Closed **nb should be a list subclass** #59

JosiahParry opened this issue on May 23, 2021 · 10 comments

One would expect that the output of the `poly2nb()` would be able to be cast as a column in a dataframe & sf object. However, since it lacks the explicit `list` class attribute, it cannot be. This would be helpful for creating spatially lagged variables in a simpler / more stream lined manner.

```
 library(spdep)
library(tidyverse)

columbus <- st_read(system.file("shapes/columbus.shp", package="spData")[1], quiet=TRUE)

# I'd anticipate this to work because poly2nb returns a list
columbus %>%
  mutate(nb = poly2nb(.),
         inc_lag = lag.list(nb2listw(nb), INC))
#> Error: Problem with `mutate()` input `nb`.
#> x Input `nb` must be a vector, not a `nb` object.
#> ℹ Input `nb` is `poly2nb(.)`.
```

{sfdep}

A tidy-ish interface to spdep for spatial dependence.

{sfdep} principles

- Always use **sf objects** for geometry
- Always return **dplyr friendly objects**
  - lists, data frames, or vectors
- Functionality is *not dependent upon dplyr*
- All functionality is **implemented using spdep**
  - nb and listw class objects when possible
- Minimal light dependencies

Making neighbors

- `st_contiguities(geometry, queen = TRUE)`
- Takes an sfc class object
  - the geometry column of an sf object
- Returns a nb class object (list)

Making neighbors

```
library(sfdep)
library(dplyr)

guerry |>
  transmute(nb = st_contiguity(geometry))
```

```
#> Simple feature collection with 85 features and 1 field
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 47680 ymin: 1703258 xmax: 1031401 ymax: 2677441
#> CRS:           NA
#> # A tibble: 85 × 2
#>    nb                                                           geometry
#>  * <nb>                                                    <MULTIPOLYGON>
#>  1 <int [4]> (((801150 2092615, 800669 2093190, 800688 2095430, 800780 2095795,...
#>  2 <int [6]> (((729326 2521619, 729320 2521230, 729280 2518544, 728751 2517520,...
#>  3 <int [6]> (((710830 2137350, 711746 2136617, 712430 2135212, 712070 2134132,...
```

The neighbor list

- Each element is an integer vector
- Elements contain row position of neighbors

```
guerry$nb[1:3]
#> [[1]]
#> [1] 36 37 67 69
#> [[2]]
#> [1]  7 49 57 58 73 76
#> [[3]]
#> [1] 17 21 40 56 61 69
```

Other neighbors

- K-Nearest Neighbor
- Distance Band
- Block Neighbors
- Custom weights using neighbor set operations
  - Union, intersection, and difference

Spatial Weights

- `st_weights(nb)` requires a nb object
  - Row standardized by default
  - Each weight is the same
- Each element is a numeric vector
  - Contains weight for each indexed observation in nb

Spatial Weights

```
guerry_nb <- guerry_nb |>
  mutate(nb = st_contiguity(geometry),
         wt = st_weights(nb))

pull(guerry_nb, "wt")[1:5]

#> [[1]]
#> [1] 0.25 0.25 0.25 0.25
#> [[2]]
#> [1] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
#> [[3]]
#> [1] 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667 0.1666667
#> [[4]]
#> [1] 0.25 0.25 0.25 0.25
#> [[5]]
#> [1] 0.3333333 0.3333333 0.3333333
```

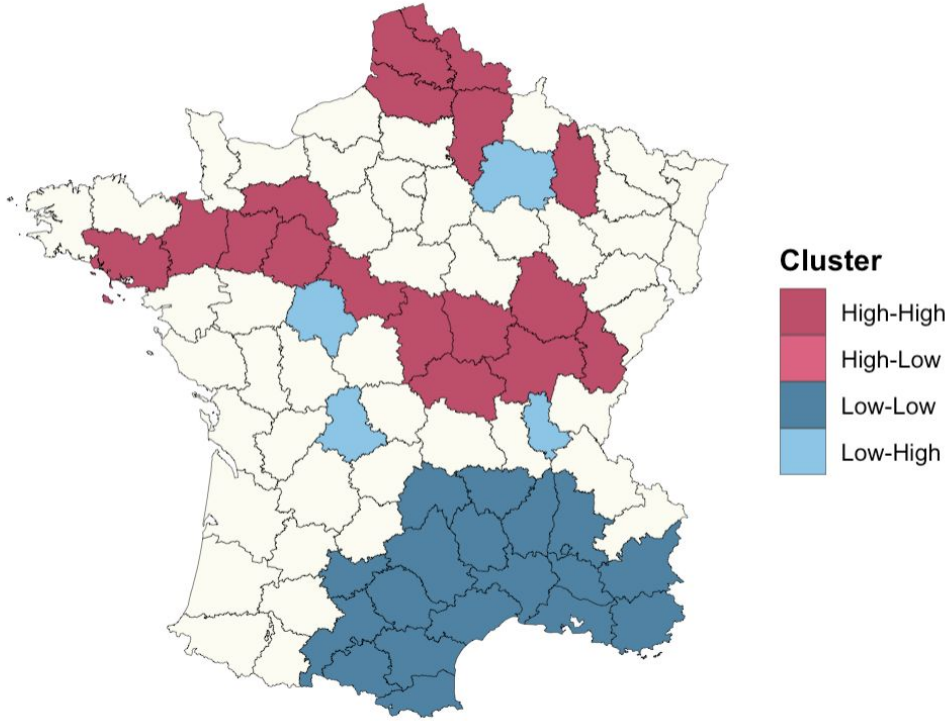
Other weights

- Distance band
  - st_nb_dists()
- Inverse distance
  - st_inverse_distance()
- Kernel weights
  - st_kernel_weights()

Comparing syntax

Clusters of Crime

France, 1833



spdep

```
library(spdep)

guerry <- sfdep::guerry
nb <- poly2nb(guerry)
listw <- nb2listw(nb)
lm_sp <- localmoran_perm(guerry$crime_pers, listw, nsim = 199)
quads <- attr(lm_sp, "quadr")
res <- cbind(guerry, lm_sp, quads)

#> Simple feature collection with 6 features and 38 fields
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 595532 ymin: 1858801 xmax: 975716 ymax: 2564568
#> CRS:           NA
#>   code_dept count ave_id_geo dept region   department crime_pers crime_prop
#> 1        01     1         49    1      E          Ain      28870      15890
#> 2        02     1        812    2      N        Aisne      26226       5521
```

sfdep

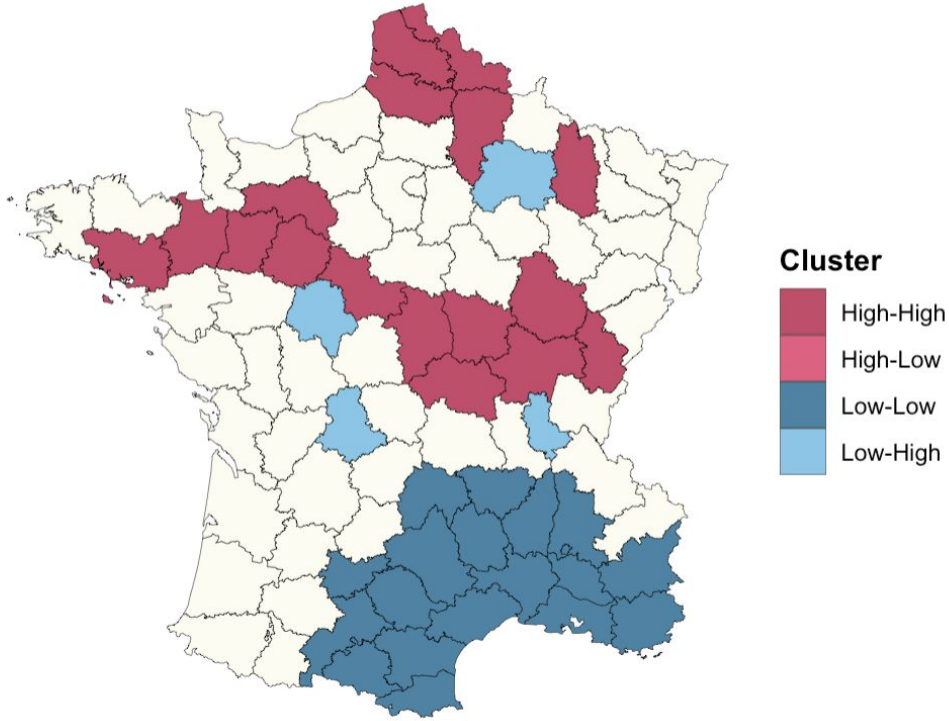
```
library(sfdep)

res <- guerry |>
  dplyr::mutate(nb = st_contiguity(geometry),
                wt = st_weights(nb),
                lm = local_moran(crime_pers, nb, wt)) |>
  tidyr::unnest(lm)
head(res)
#> Simple feature collection with 6 features and 40 fields
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 595532 ymin: 1858801 xmax: 975716 ymax: 2564568
#> CRS:           NA
#> # A tibble: 6 × 41
#>   code_dept count ave_id_geo  dept region department   crime_pers crime_prop
#>   <fct>     <dbl>      <dbl> <int> <fct>  <fct>             <int>      <int>
#> 1 01            1         49     1 E      Ain               28870      15890
#> 2 02            1        812     2 N      Aisne             26226       5521
```

The map

```
res |>
  dplyr::mutate(Cluster = ifelse(p_folded_sim <= 0.1, as.character(pysal), NA)) |>
  ggplot(aes(fill = Cluster)) +
  geom_sf(lwd = 0.1, color = "black") +
  scale_fill_manual(
    values = c("High-High" = "#bd4f6b",
               "High-Low" = "#db6381",
               "Low-Low" = "#5084a3",
               "Low-High" = "#8cc5e6"),
    na.value = "#fcfcf2"
    ) +
  theme_void() +
  labs(title = "Clusters of Crime",
       subtitle = "France, 1833") +
    theme(title = element_text(face = "bold"))
```

Clusters of Crime

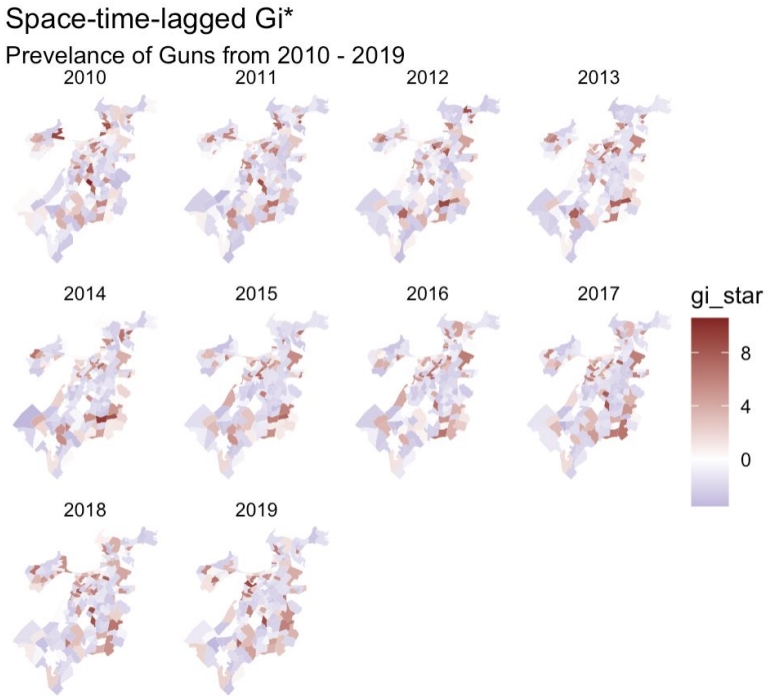
France, 1833

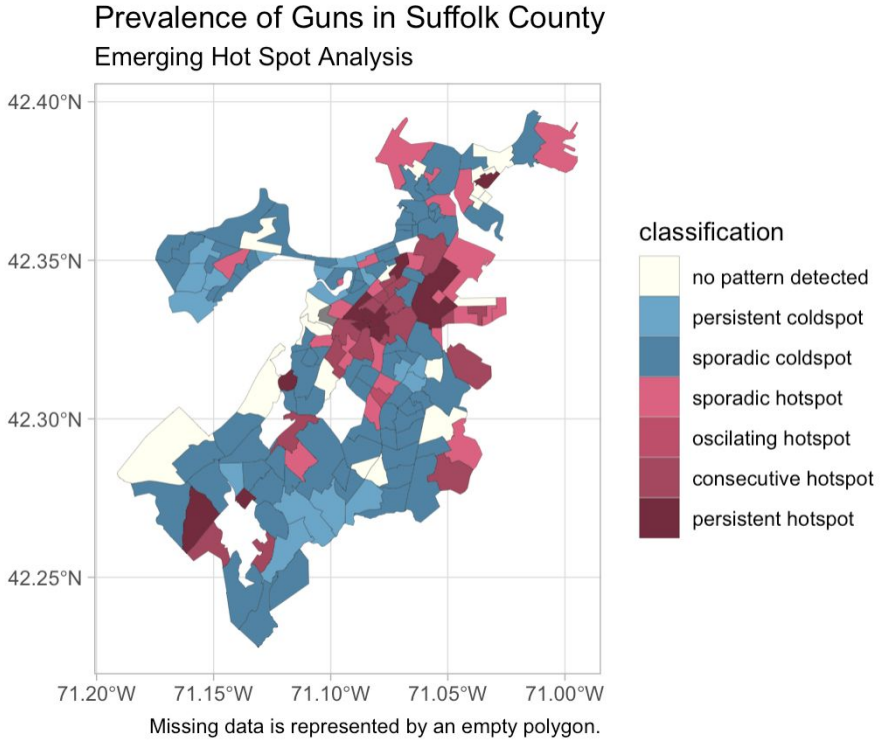


Extending spdep

- Local Join counts
- Colocation Quotients
- Emerging hot spot analysis
  - Linked data and geometry spacetime objects
- Point pattern centrography
  - Functionality not available in spatstat
- Casting to {sfnetworks}

Extending spdep:
Emerging Hot Spot Analysis

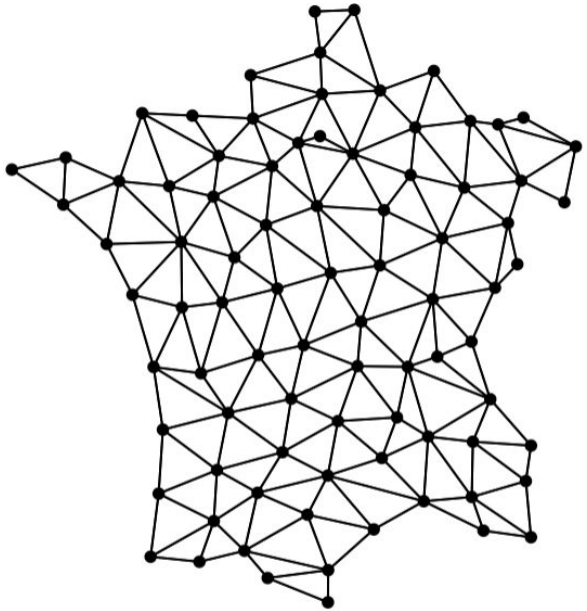




Integration with sfnetworks

- Neighbors are nodes
- Weights are edges

```
library(sfdep)
library(dplyr)

g_ntwk <- guerry |>
  mutate(nb = st_contiguity(geometry),
         wt = st_weights(nb)) |>
  st_as_graph()

plot(g_ntwk)
```



What's next?

- User validation
- Ensure R parity with libpysal
  - sfdep has parity with pysal ESDA
- Ease adoption with recordings and vignettes
- PRs to spdep from extended statistics
- Make a hex logo

Give it a go

JosiahParry/**sfdep**

A tidy interface for spatial dependence

1 Contributor

10 Issues

42 Stars

2 Forks