Assessing hurricane exposure with R

Packages to facilitate environmental epidemiology

Brooke Anderson, Colorado State University

Required set-up (1)

- Install R
- Install RStudio (Recommended but not required)

Required set-up (2)

Hosting Data Packages via drat: A Case Study with Hurricane Exposure Data

by G. Brooke Anderson and Dirk Eddelbuettel

Abstract Data-only packages offer a way to provide extended functionality for other R users. However, such packages can be large enough to exceed the package size limit (5 megabytes) for the Comprehensive R Archive Network (CRAN). As an alternative, large data packages can be posted to additional repostiories beyond CRAN itself in a way that allows smaller code packages on CRAN to access and use the data. The **drat** package facilitates creation and use of such alternative repositories and makes it particularly simple to host them via GitHub. CRAN packages can draw on packages posted to **drat** repositories through the use of the 'Additonal_repositories' field in the DESCRIPTION file. This paper describes how R users can create a suite of coordinated packages, in which larger data packages are hosted in an alternative repository created with **drat**, while a smaller code package that interacts with this data is created that can be submitted to CRAN.

Required set-up (2)

- Install `hurricaneexposuredata` (from our personal repo)

```
install.packages("drat") # Only run if you don't have `drat`
library("drat")

addRepo("geanders")
install.packages("hurricaneexposuredata")
```

Required set-up (3)

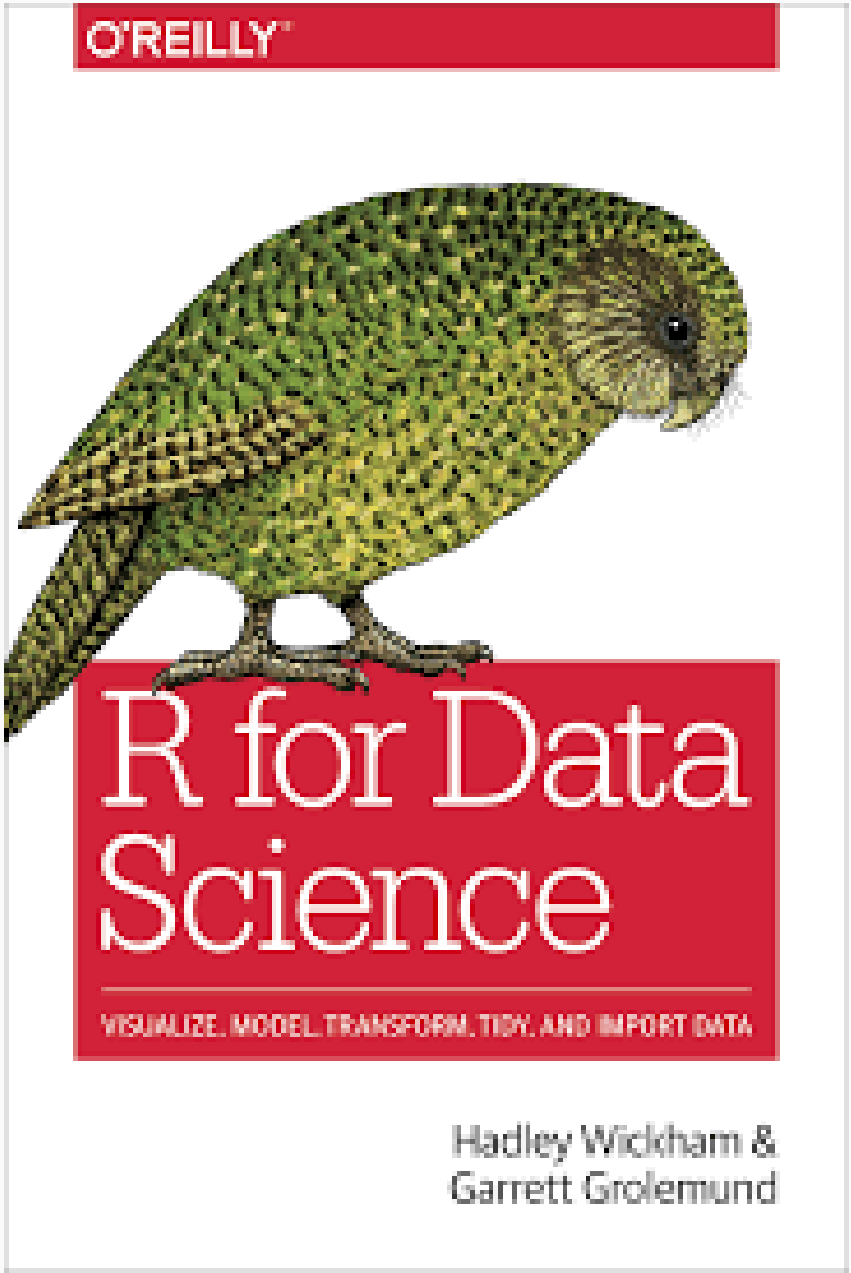
- Install `hurricaneexposure` (from GitHub)

```
install.packages("devtools") # If you don't have `devtools`
library("devtools")
install_github("geanders/hurricaneexposure")
```

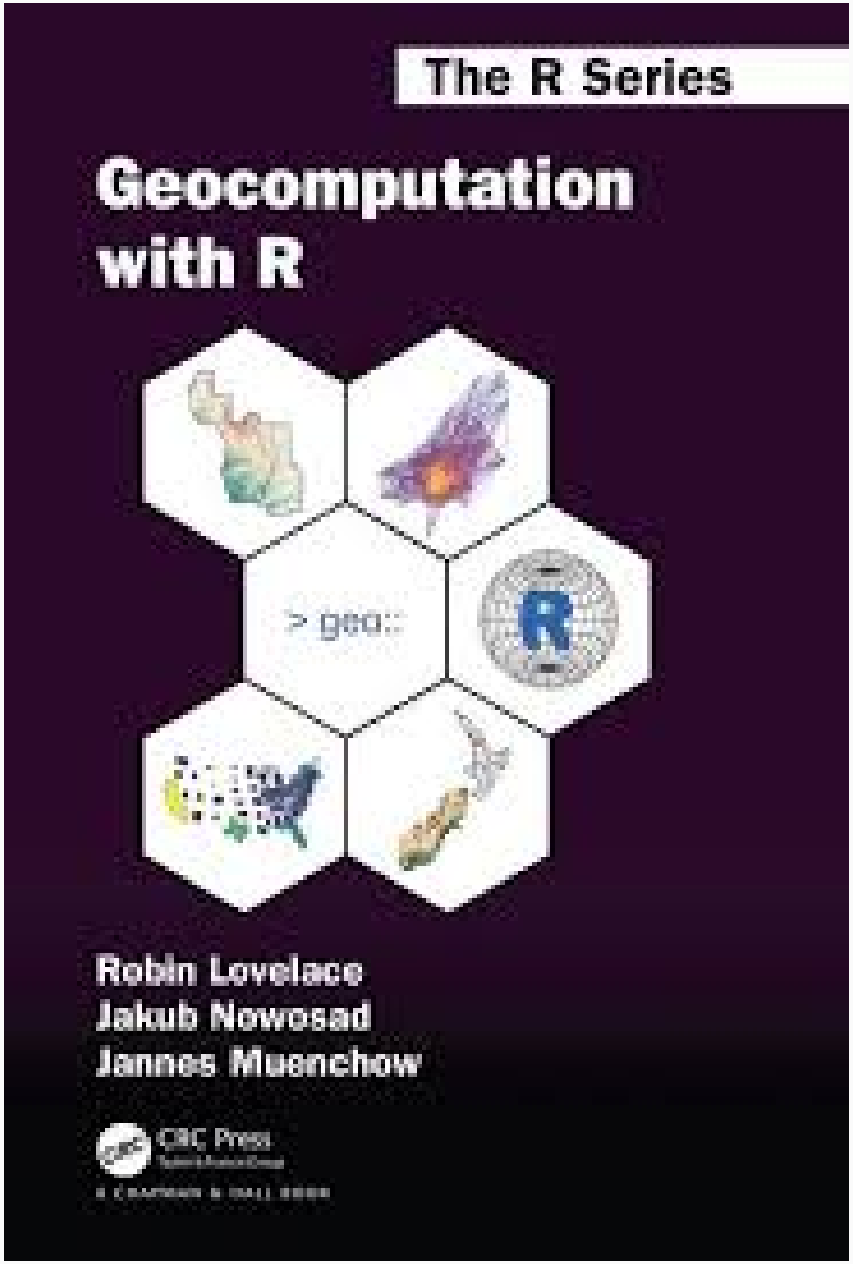
Required set-up (4)

- Install supplementary packages:
  - `tidyverse`
  - `sf`
  - `tigris`
  - `rnaturalearth` and `rnaturalearthdata`
  - `weathermetrics`
  - `noaastormevents`
  - `stormwindmodel`
  - `viridis`

For more on R


O'REILLY
R for Data Science
VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA
Hadley Wickham &
Garrett Grolemund

For more on geocomputation with R



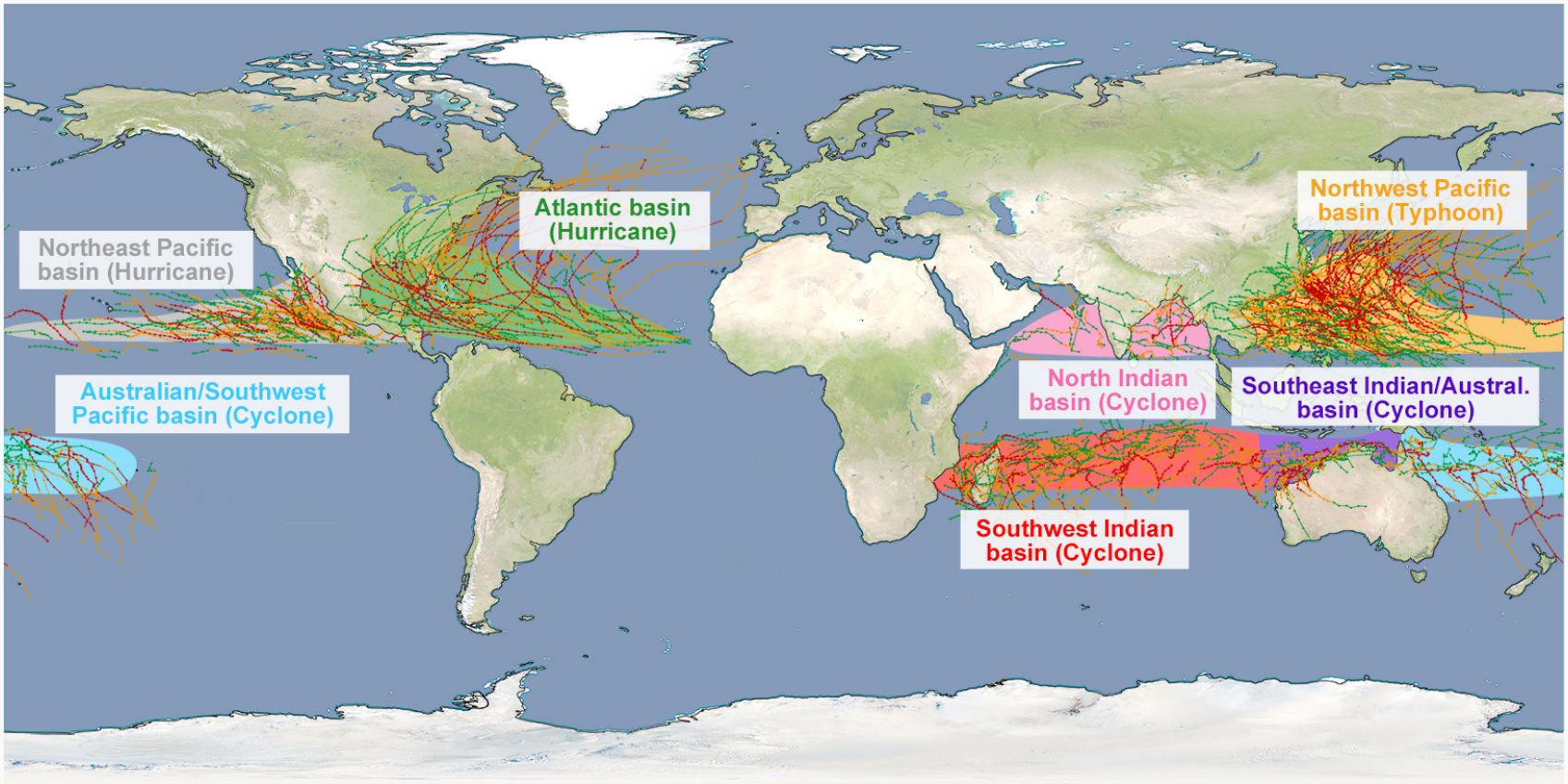
Load packages

To start, load only the two `hurricaneexposure` packages and the `tidyverse` package (for standard data science R tools).

```
library("hurricaneexposuredata")
library("hurricaneexposure")
library("tidyverse")
```

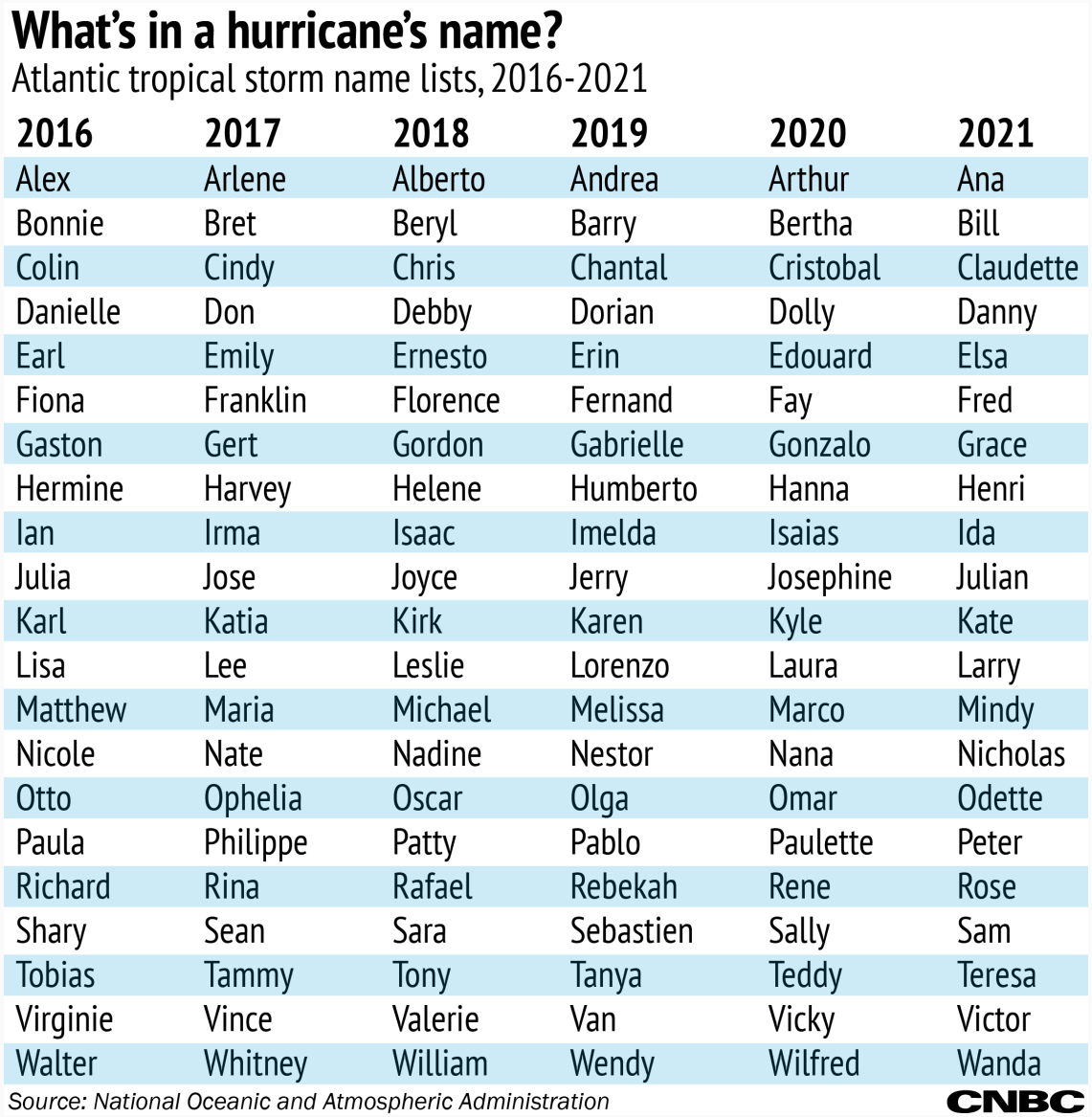
Basics on tropical cyclone data

Tropical cyclone basins



Source: Deutsche Wetterdienst

Tropical cyclone names



What's in a hurricane's name?

Atlantic tropical storm name lists, 2016-2021

2016	2017	2018	2019	2020	2021
Alex	Arlene	Alberto	Andrea	Arthur	Ana
Bonnie	Bret	Beryl	Barry	Bertha	Bill
Colin	Cindy	Chris	Chantal	Cristobal	Claudette
Danielle	Don	Debby	Dorian	Dolly	Danny
Earl	Emily	Ernesto	Erin	Edouard	Elsa
Fiona	Franklin	Florence	Fernand	Fay	Fred
Gaston	Gert	Gordon	Gabrielle	Gonzalo	Grace
Hermine	Harvey	Helene	Humberto	Hanna	Henri
Ian	Irma	Isaac	Imelda	Isaias	Ida
Julia	Jose	Joyce	Jerry	Josephine	Julian
Karl	Katia	Kirk	Karen	Kyle	Kate
Lisa	Lee	Leslie	Lorenzo	Laura	Larry
Matthew	Maria	Michael	Melissa	Marco	Mindy
Nicole	Nate	Nadine	Nestor	Nana	Nicholas
Otto	Ophelia	Oscar	Olga	Omar	Odette
Paula	Philippe	Patty	Pablo	Paulette	Peter
Richard	Rina	Rafael	Rebekah	Rene	Rose
Shary	Sean	Sara	Sebastien	Sally	Sam
Tobias	Tammy	Tony	Tanya	Teddy	Teresa
Virginie	Vince	Valerie	Van	Vicky	Victor
Walter	Whitney	William	Wendy	Wilfred	Wanda

Source: National Oceanic and Atmospheric Administration

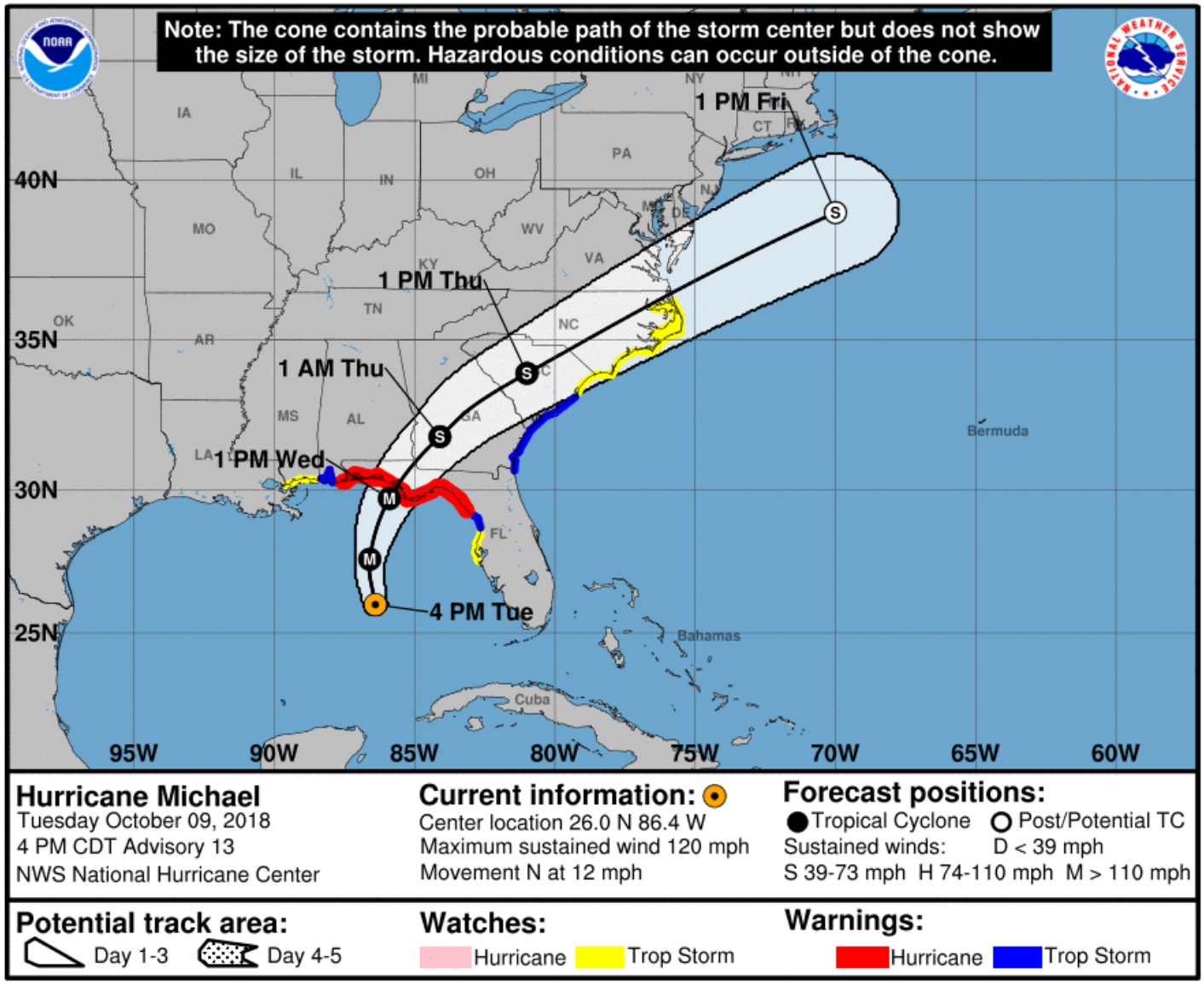
CNBC

Source: CNBC / NOAA

Retiring names

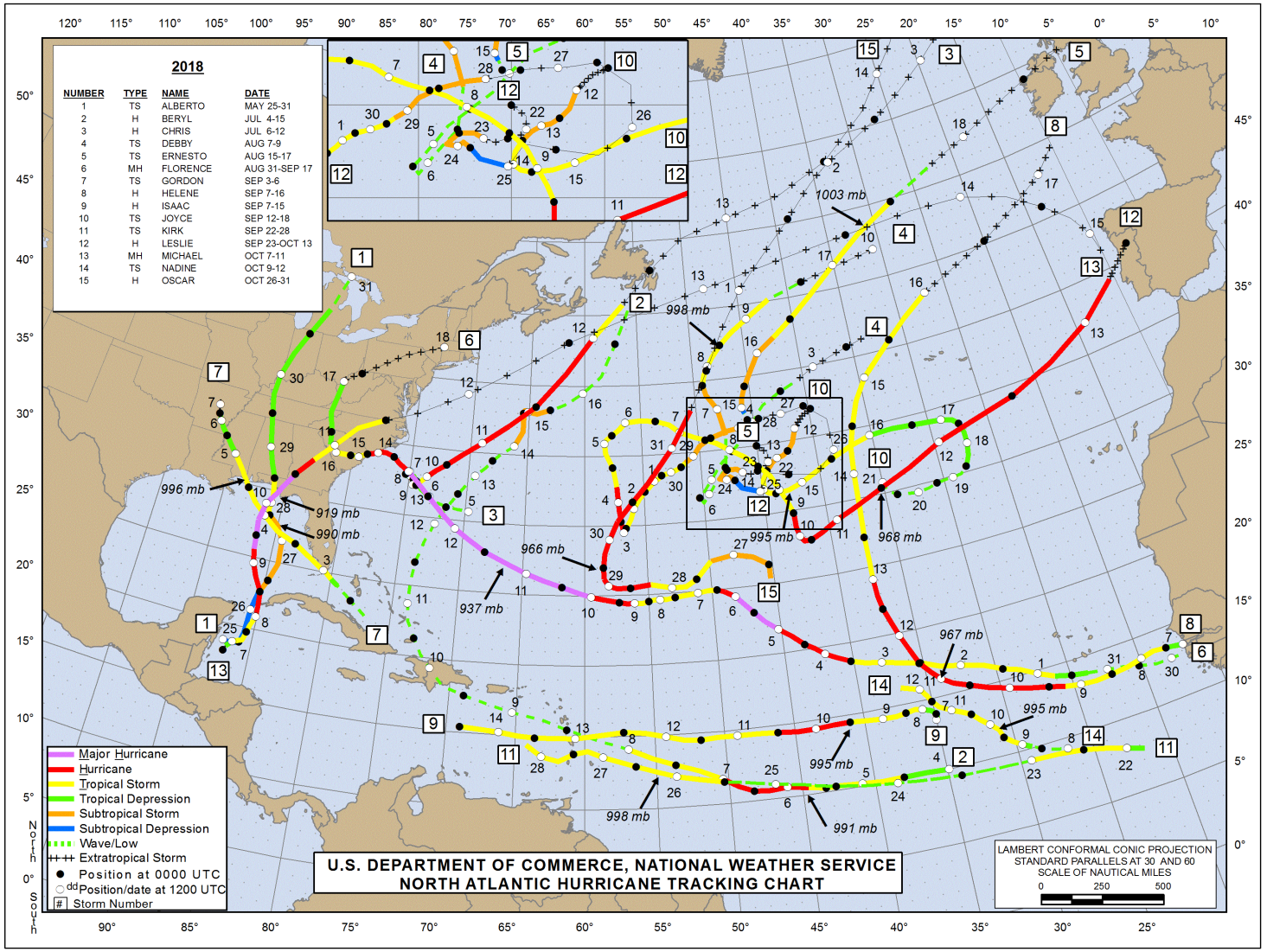
Source: Lex18

Storm track data



Source: NOAA

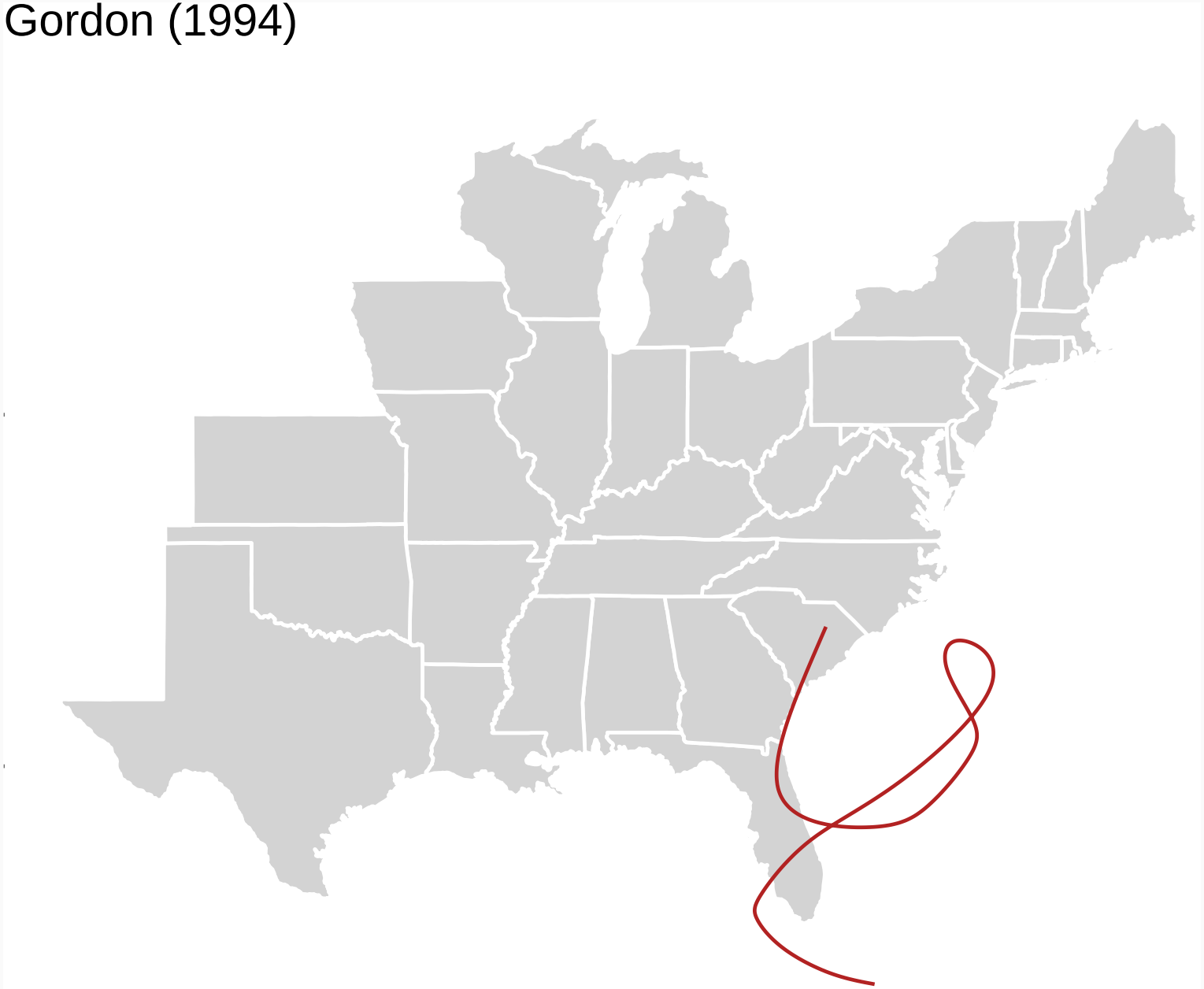
“Best tracks” data



Source: NOAA

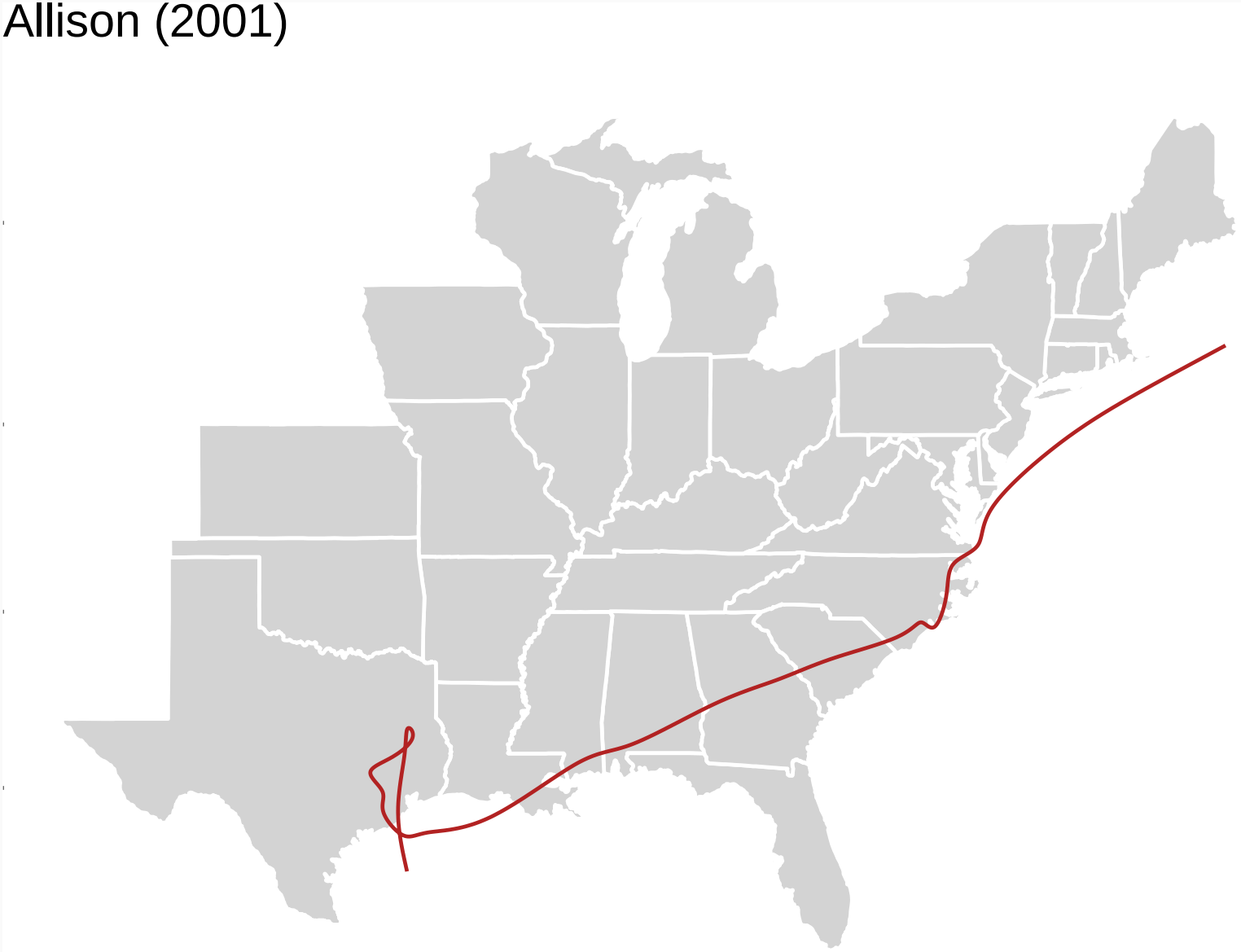
Unusual storm tracks

Gordon (1994)



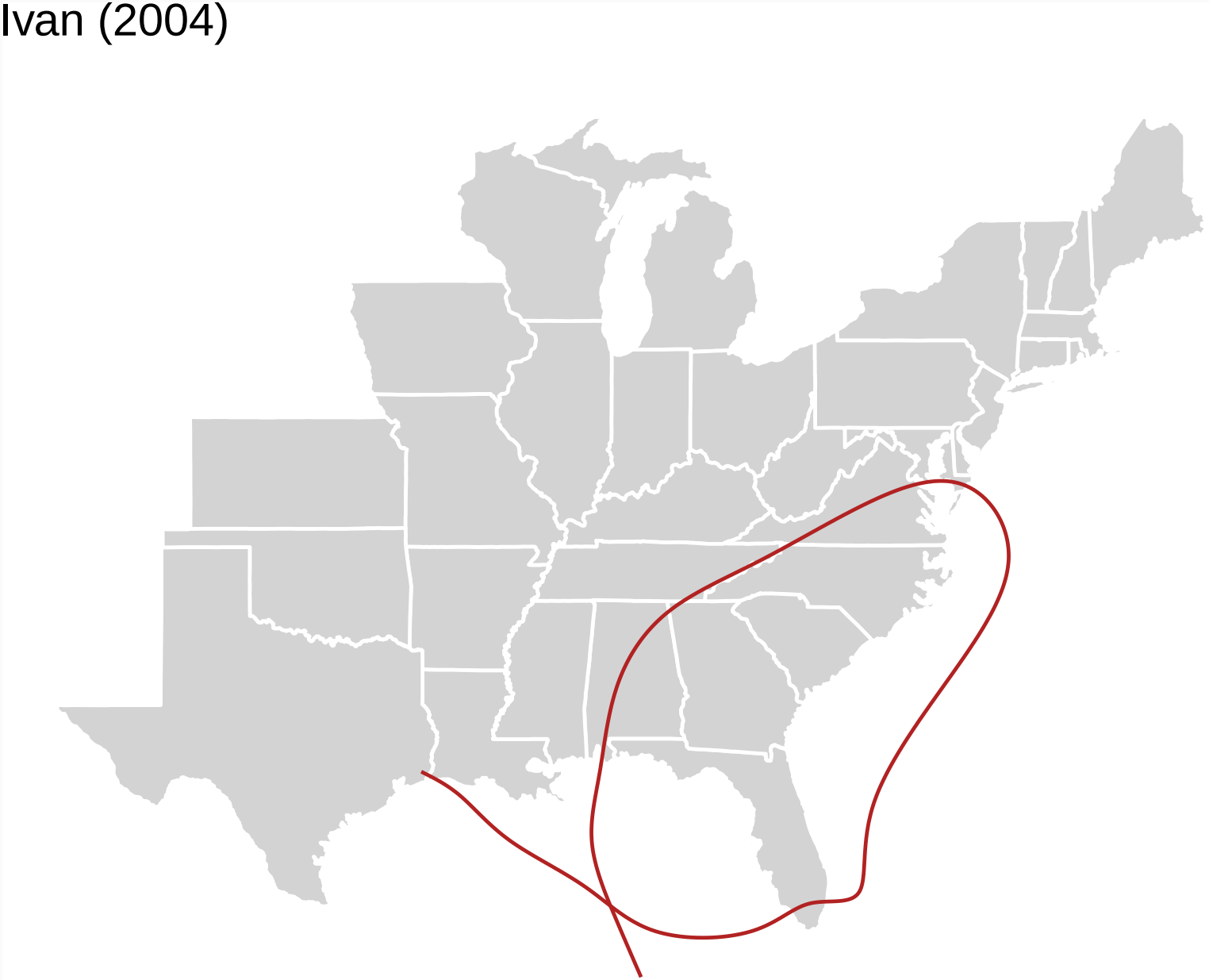
Unusual storm tracks

Allison (2001)

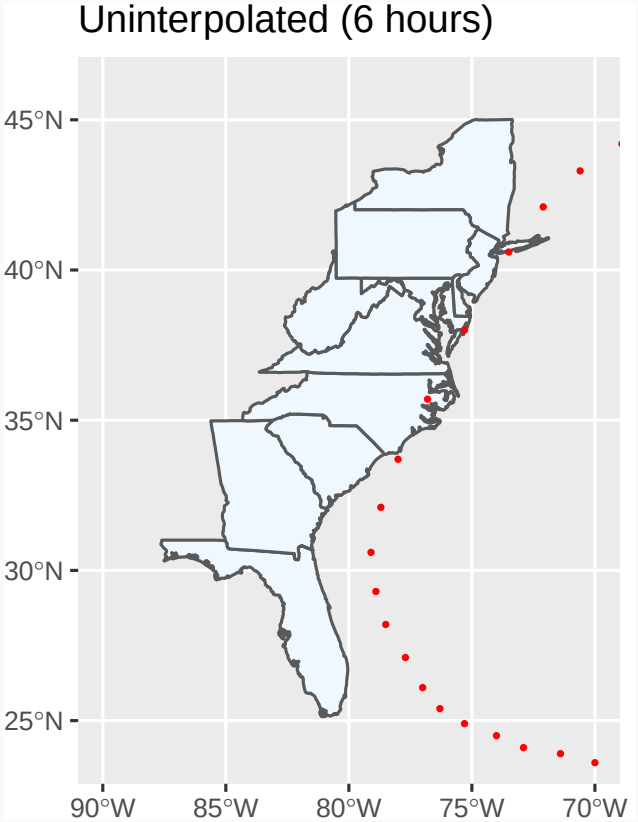


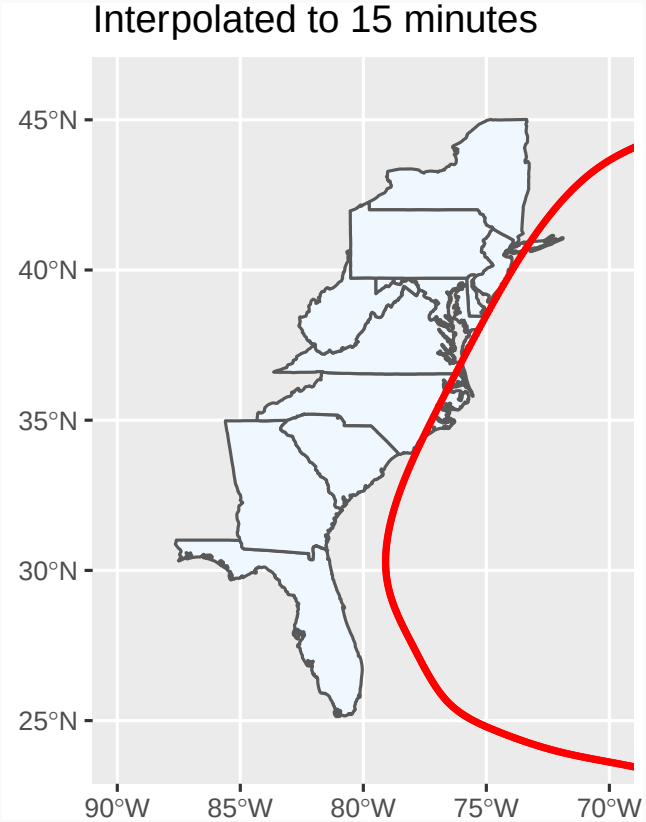
Unusual storm tracks

Ivan (2004)



Interpolating tracks

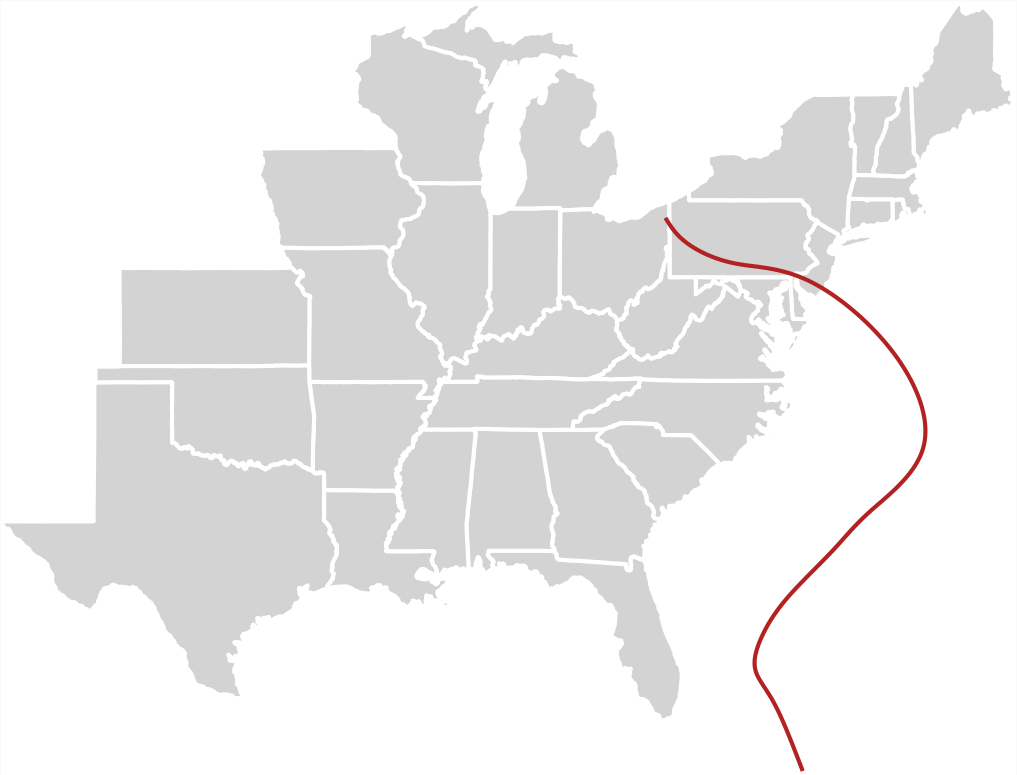




Mapping storm tracks

You can map a storm's track with the `map_tracks` function from `hurricaneexposure`. For example, map the track of Hurricane Sandy with:

```
map_tracks(storm = "Sandy-2012")
```



Mapping storm tracks

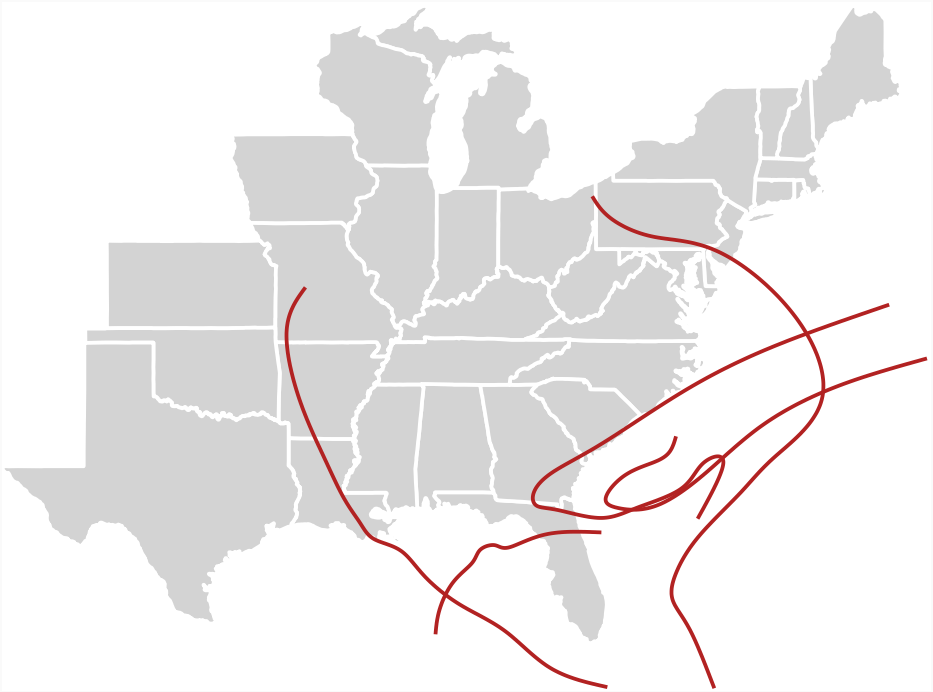
The output is a ggplot-class object, so you can add elements as with other ggplot-class objects:

```
map_tracks(storm = "Sandy-2012") +
   labs(title = "Central track of Hurricane Sandy, 2012",
        caption = "Based on HURDAT2 Best Tracks")
```

Mapping storm tracks

You can plot the tracks of multiple storms. For example, you can plot the tracks of all the tracked storms that came near the US in 2012 with:

```
map_tracks(storm = c("Alberto-2012", "Beryl-2012", "Debby-2012",
                     "Isaac-2012", "Sandy-2012"))
```



Storm track data

This function is using storm track data that we've included in the `hurricaneexposuredata` package, in a dataset called `hurr_tracks`:

```
data("hurr_tracks")
hurr_tracks %>% sample_n(3)
```

```
##     storm_id         date latitude longitude wind
## 1 Chris-2006 200608040600     20.8     -70.2   25
## 2   Ike-2008 200809101200     23.8     -85.2   80
## 3 Mitch-1998 199811031200     19.4     -92.1   25
```

Storm track data

You can work directly with this dataset. For example, you can use regular expressions to identify all the storms in the dataset within a certain year and the number of tracking data points for each in this data:

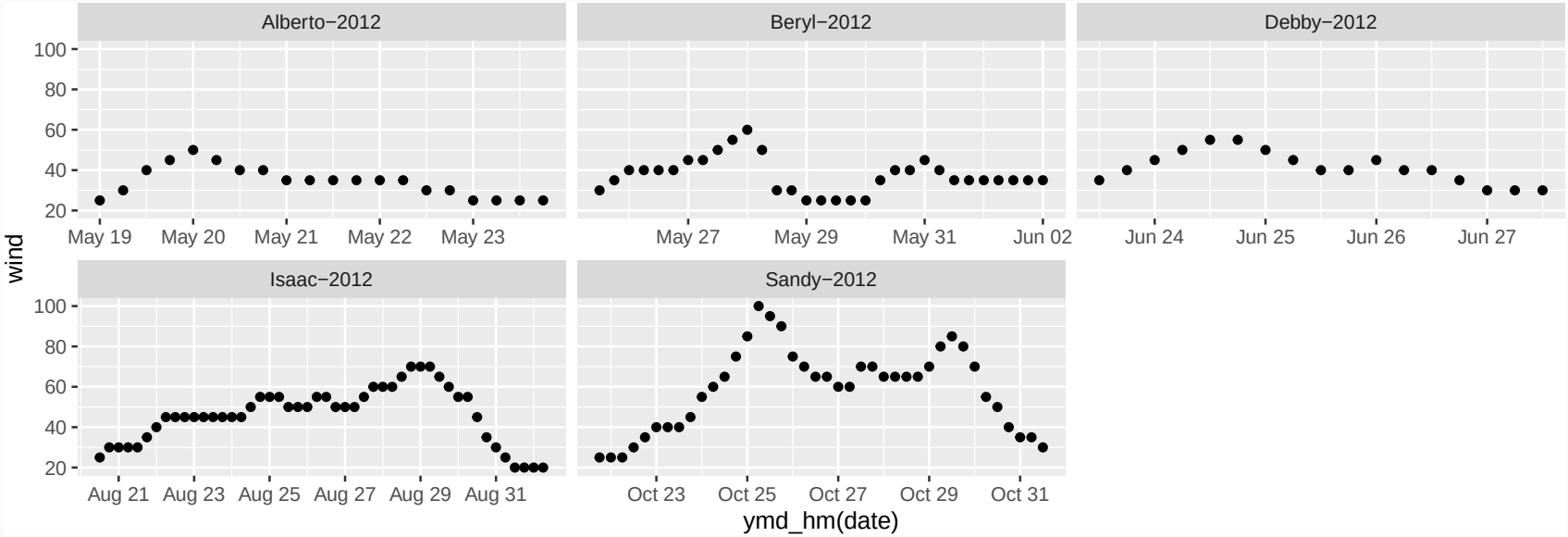
```
hurr_tracks %>%
   filter(str_detect(storm_id, "2012")) %>%
   group_by(storm_id) %>%
   summarize(n = n(), max_wind = max(wind),
             start_date = first(date))
```

```
## # A tibble: 5 x 4
##   storm_id          n max_wind start_date
##   <chr>         <int>    <dbl> <chr>
## 1 Alberto-2012     20       50 201205190000
## 2 Beryl-2012       31       60 201205251200
## 3 Debby-2012       17       55 201206231200
## 4 Isaac-2012       48       70 201208201200
## 5 Sandy-2012       40      100 201210211800
```

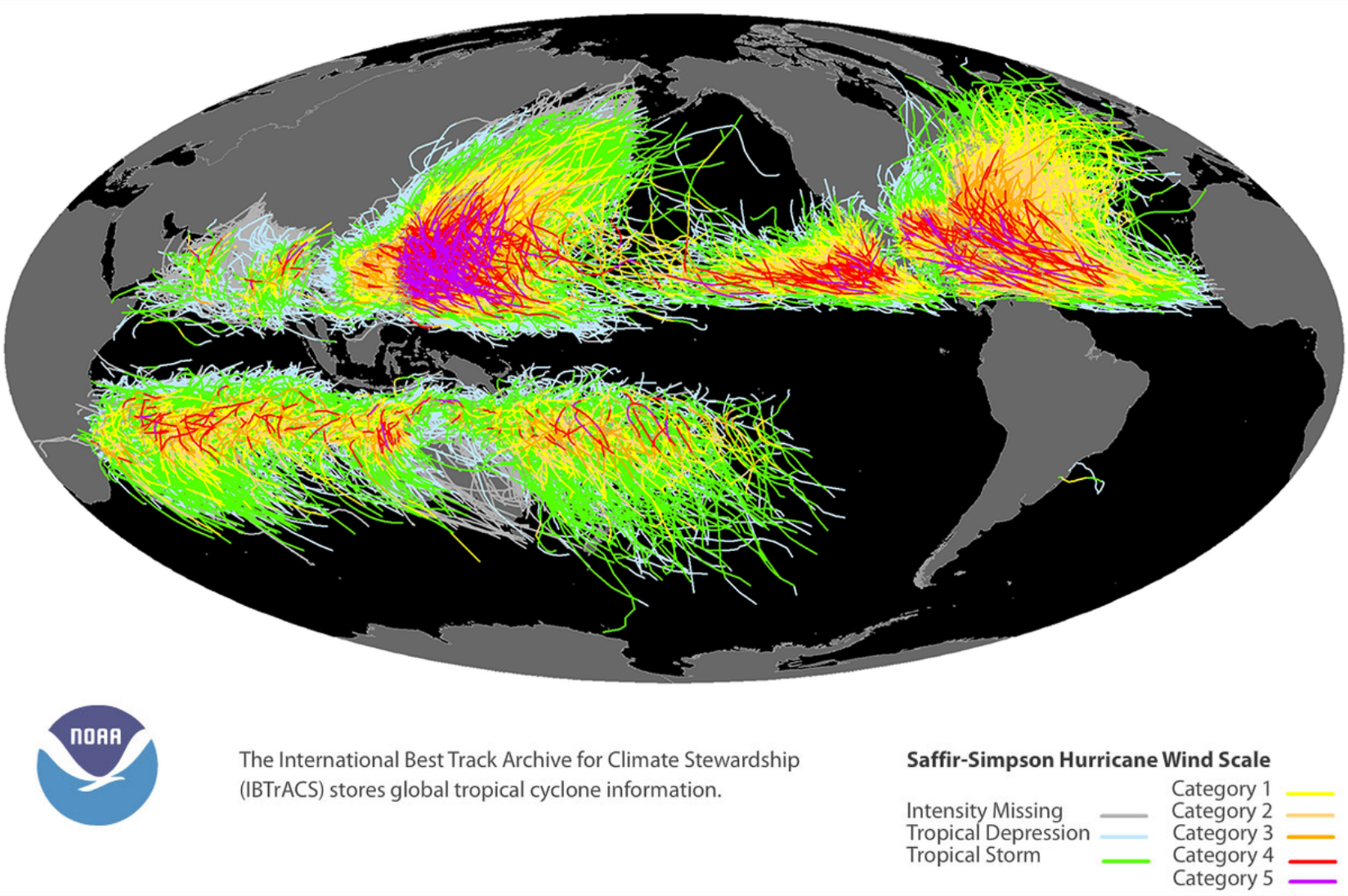
Storm track data

Similarly, you can explore patterns in the storm track measurements, like each storm's pattern of maximum sustained winds over time:

```
library("lubridate")
hurr_tracks %>%
    filter(str_detect(storm_id, "2012")) %>%
    ggplot(aes(x = ymd_hm(date), y = wind)) +
    geom_point() +
    facet_wrap(~ storm_id, scales = "free_x")
```



Other R sources for storm tracks



The R package `noaastorms` wraps an API for IBTrACS.

Your turn

- Map the tracks of one of the unusual storms using `map_tracks`.
- Map the tracks of several of the unusual storms on the same map using `map_tracks`

R tools for geospatial tools

sf framework for geospatial data

The sf package allows you to work with geospatial data within the "tidyverse" framework. You can convert a regular dataframe to an sf object using st_as_sf:

```
library("sf")
storms_2012 <- hurr_tracks %>%
   filter(str_detect(storm_id, "2012")) %>%
   st_as_sf(coords = c("longitude", "latitude"))
```

sf framework for geospatial data

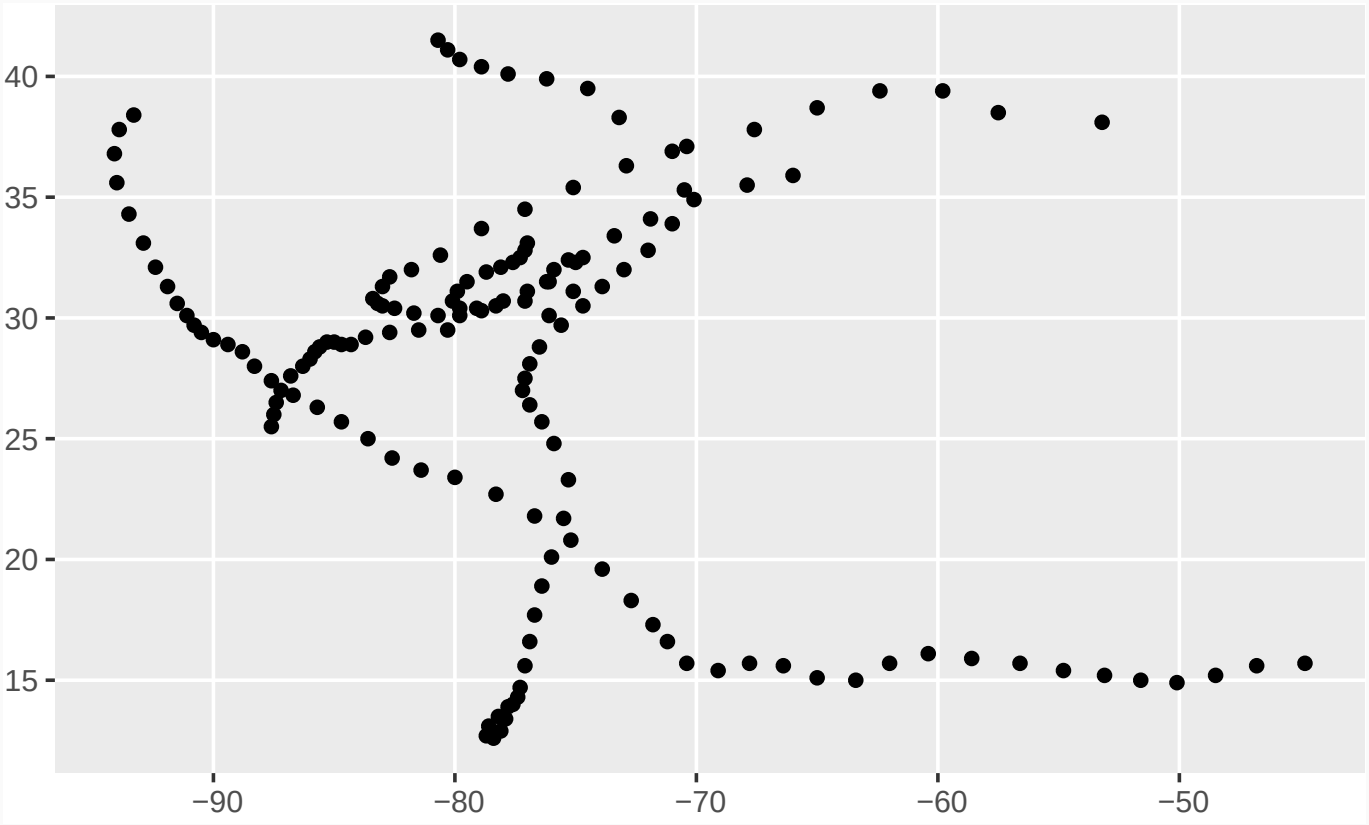
```
storms_2012
```

```
## Simple feature collection with 156 features and 3 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -94.1 ymin: 12.6 xmax: -44.8 ymax: 41.5
## epsg (SRID):    NA
## proj4string:    NA
## First 10 features:
##        storm_id         date wind           geometry
## 1  Alberto-2012 201205190000   25   POINT (-77 33.1)
## 2  Alberto-2012 201205190600   30 POINT (-77.1 32.8)
## 3  Alberto-2012 201205191200   40 POINT (-77.3 32.5)
## 4  Alberto-2012 201205191800   45 POINT (-77.6 32.3)
## 5  Alberto-2012 201205200000   50 POINT (-78.1 32.1)
## 6  Alberto-2012 201205200600   45 POINT (-78.7 31.9)
## 7  Alberto-2012 201205201200   40 POINT (-79.5 31.5)
## 8  Alberto-2012 201205201800   40 POINT (-79.9 31.1)
## 9  Alberto-2012 201205210000   35 POINT (-80.1 30.7)
```

sf framework for geospatial data

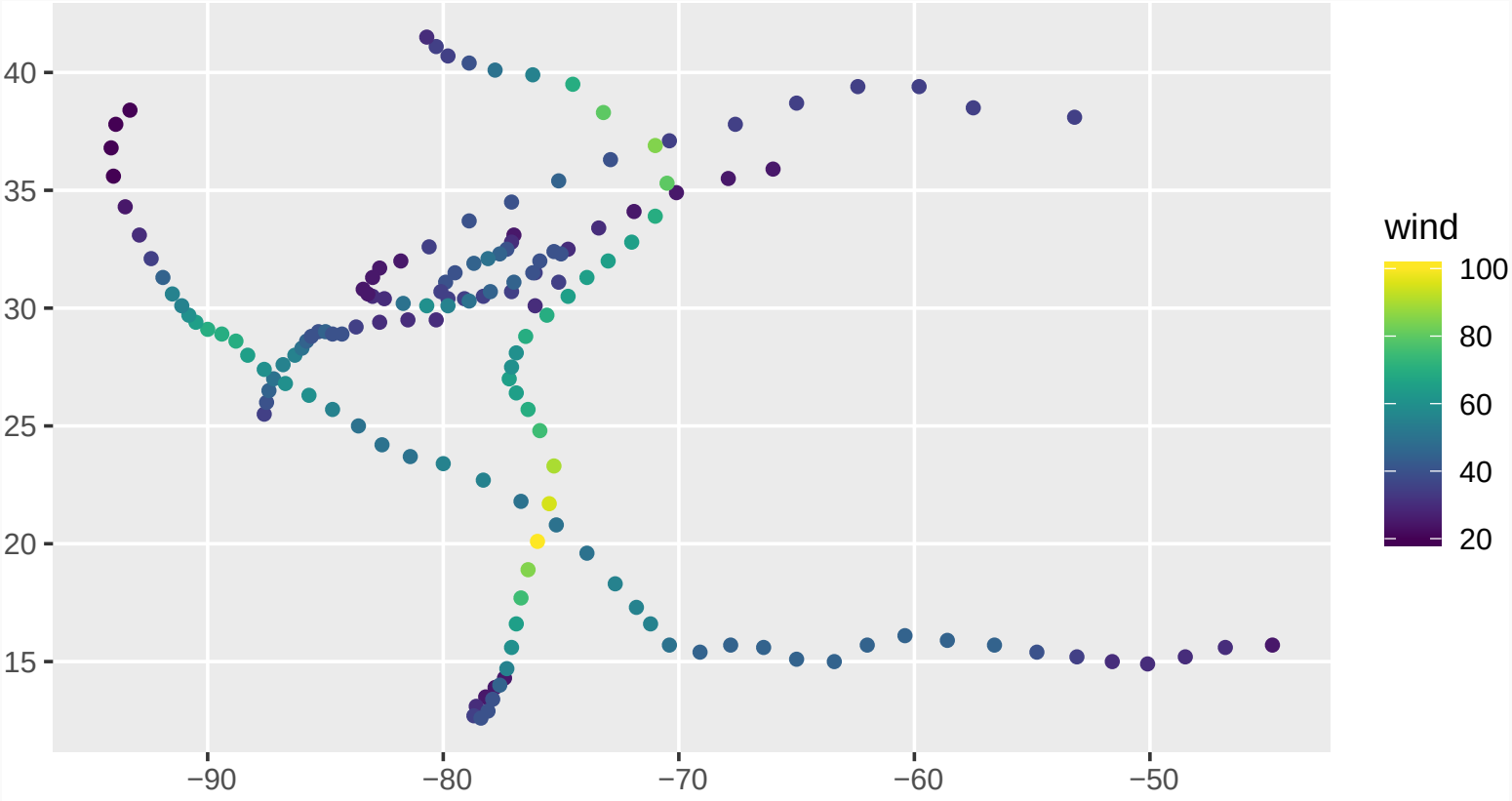
You can use ggplot2 to create maps with sf objects:

```
ggplot() +
    geom_sf(data = storms_2012)
```



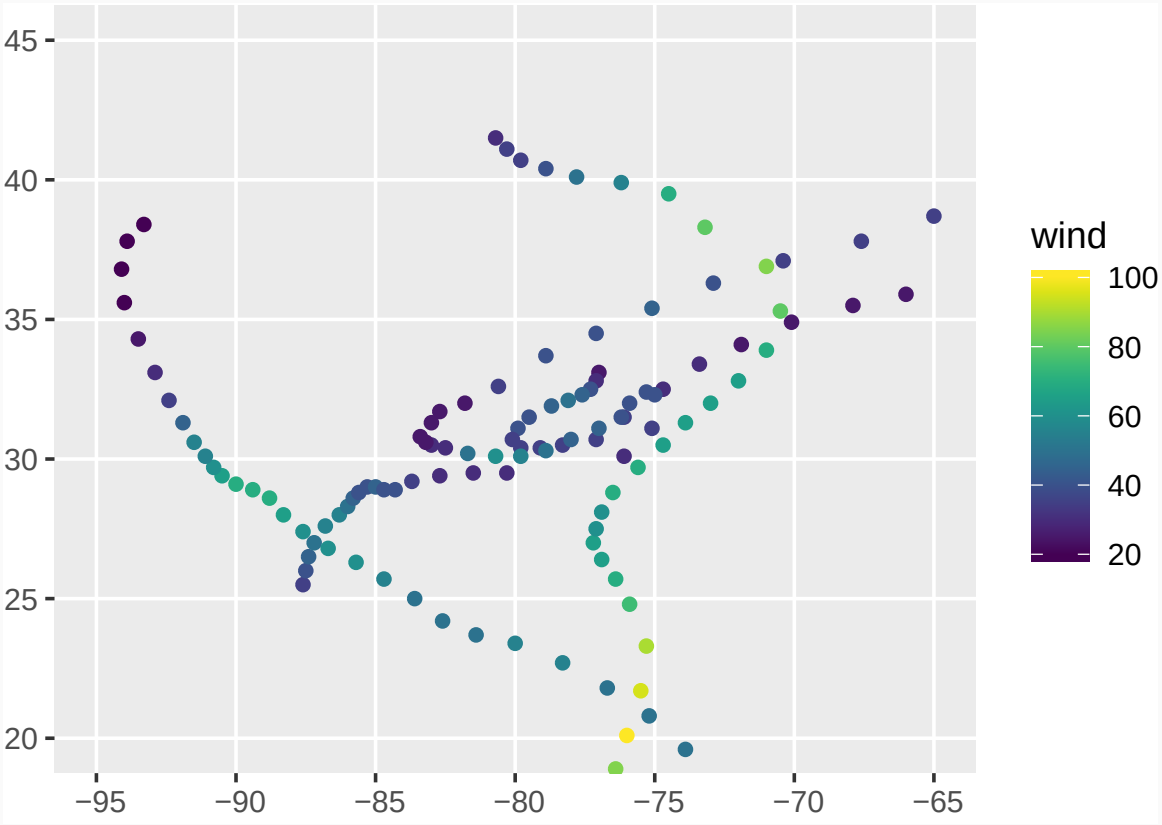
sf framework for geospatial data

```
library("viridis")
ggplot() +
    geom_sf(data = storms_2012,
            aes(color = wind)) +
    scale_color_viridis()
```



sf framework for geospatial data

```
ggplot() +
    geom_sf(data = storms_2012,
            aes(color = wind)) +
    scale_color_viridis() +
    xlim(c(-95, -65)) + ylim(c(20, 45))
```



`tigris` package

The `tigris` package allows you to access spatial data from the US Census directly from R. For example, to pull geographic data for county boundaries for Florida counties, run:

```
library("tigris")
ny_counties <- counties(state = "NY",
                        cb = FALSE, resolution = "20m",
                        class = "sf")
```

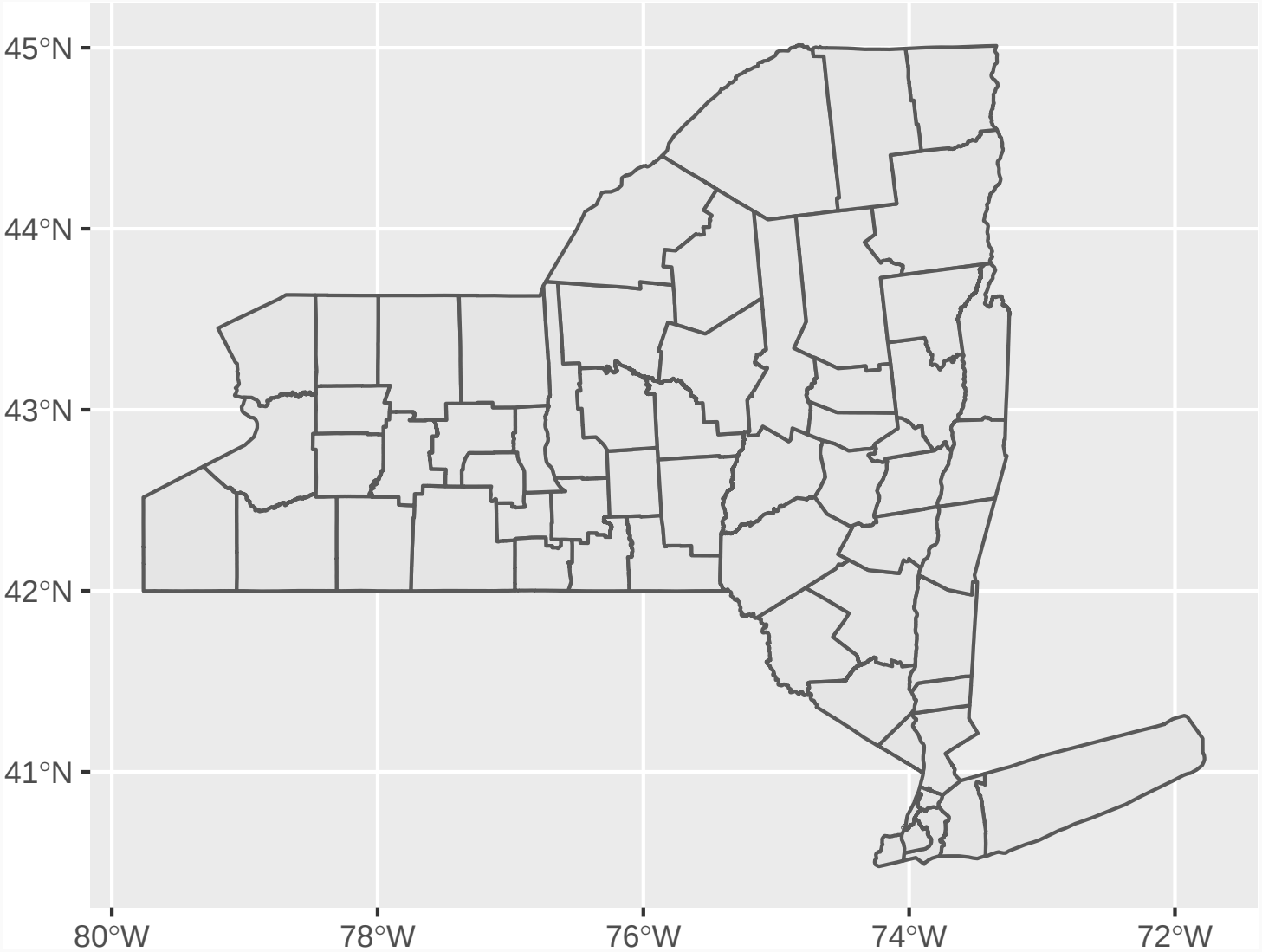
tigris package

```
ny_counties %>% slice(1:3)
```

```
## Simple feature collection with 3 features and 17 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -78.30932 ymin: 41.99826 xmax: -73.57334 ymax: 43.3980
## epsg (SRID):    4269
## proj4string:    +proj=longlat +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +no_defs
##   STATEFP COUNTYFP COUNTYNS GEOID     NAME        NAMELSAD LSAD CLASSFP
## 1      36      101 00974148 36101  Steuben  Steuben County   06      H1
## 2      36      091 00974143 36091 Saratoga Saratoga County   06      H1
## 3      36      003 00974100 36003 Allegany Allegany County   06      H1
##   MTFCC CSAFP CBSAFP METDIVFP FUNCSTAT      ALAND   AWATER    INTPTLAT
## 1 G4020   236  18500     <NA>        A 3601504767 35055663 +42.2667252
## 2 G4020   104  10580     <NA>        A 2097880370 87571838 +43.1061353
## 3 G4020  <NA>   <NA>     <NA>        A 2665875177 13153776 +42.2478532
##       INTPTLON                       geometry
## 1 -077.3855253 MULTIPOLYGON (((-77.42309 4...
## 2 -073.8553872 MULTIPOLYGON (((-74.12434 4...
## 3 -078.0261531 MULTIPOLYGON (((-78.30919 4...
```

tigris package

```
ggplot() +
    geom_sf(data = ny_counties)
```



Putting things together

```
sandy_2012 <- storms_2012 %>%
   st_set_crs(4269) %>%
   filter(storm_id == "Sandy-2012")
sandy_2012 %>% slice(1:3)
```

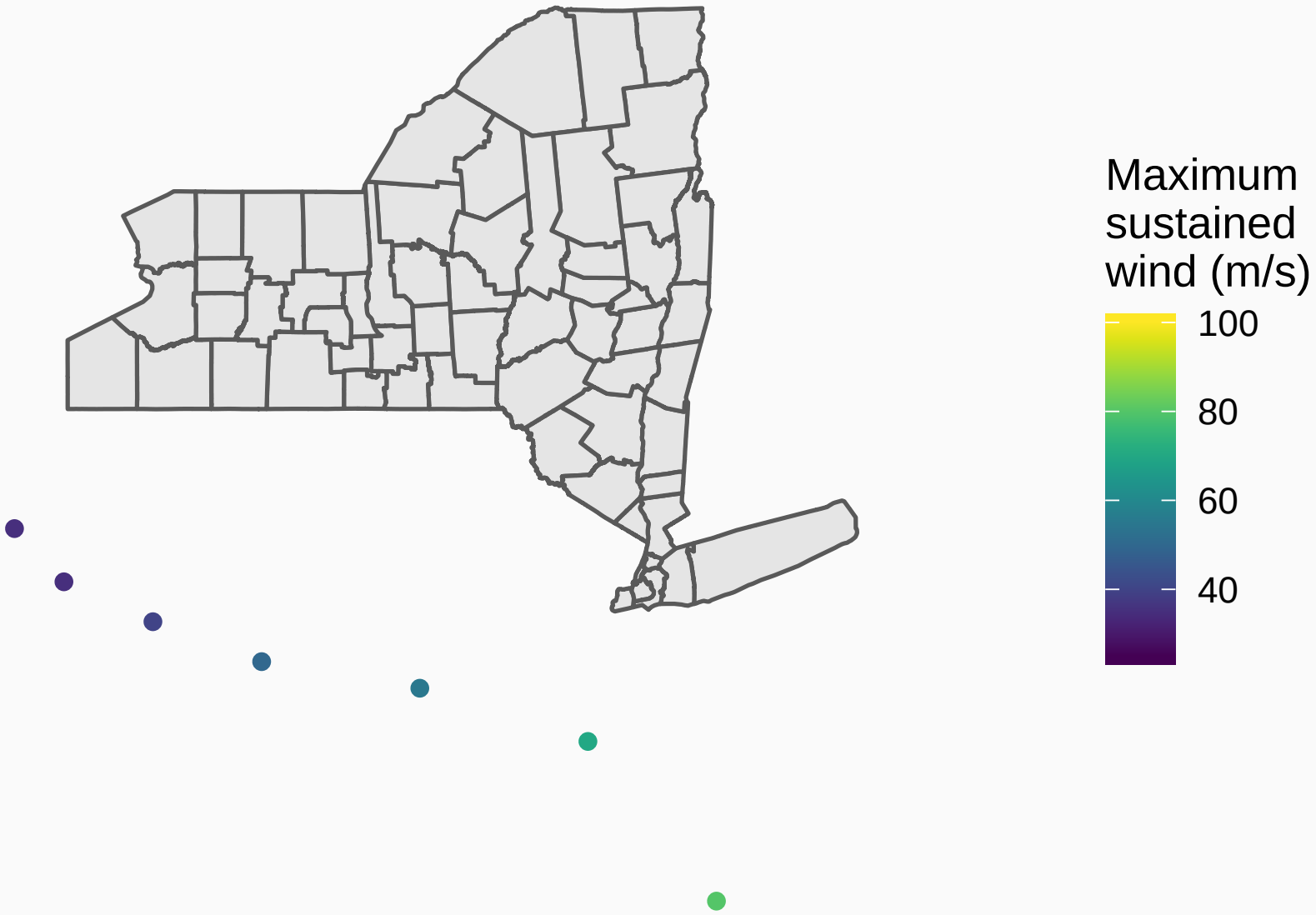
```
## Simple feature collection with 3 features and 3 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -78.2 ymin: 13.5 xmax: -77.4 ymax: 14.3
## epsg (SRID):    4269
## proj4string:    +proj=longlat +ellps=GRS80 +towgs84=0,0,0,0,0
##     storm_id          date wind           geometry
## 1 Sandy-2012 201210211800   25 POINT (-77.4 14.3)
## 2 Sandy-2012 201210220000   25 POINT (-77.8 13.9)
## 3 Sandy-2012 201210220600   25 POINT (-78.2 13.5)
```

Putting things together

You can plot several "layers" of geospatial data by adding `geom_sf` calls for the ggplot object:

```
ggplot() +
   geom_sf(data = ny_counties) +
   geom_sf(data = sandy_2012,
           aes(color = wind)) +
   scale_color_viridis() +
   xlim(c(-80, -70)) +
   ylim(c(38, 46)) +
   theme_void() +
   labs(color = "Maximum\nsustained\nwind (m/s)")
```

Putting things together

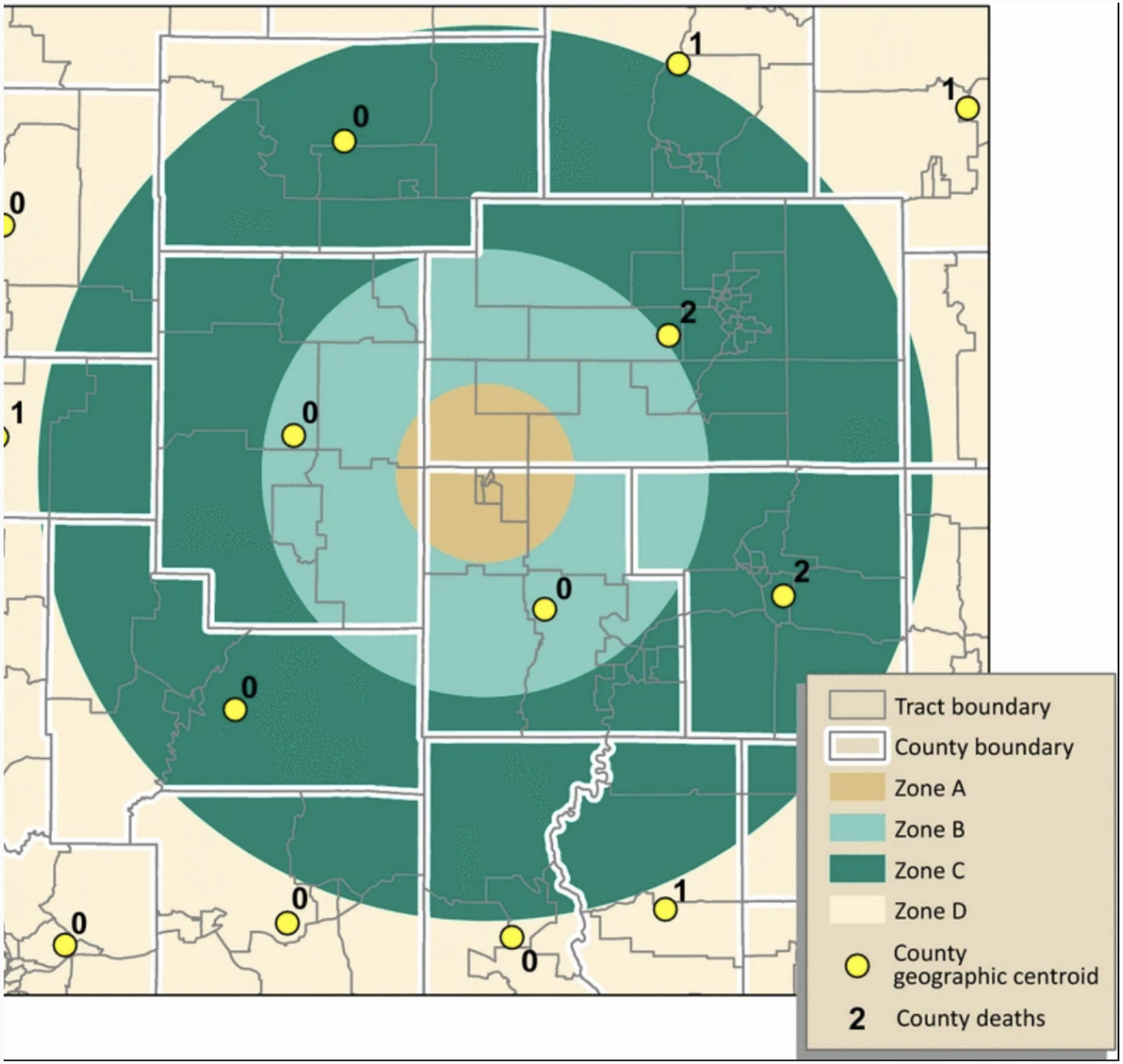


Your turn

- Use `tigris` and `sf` to create a map of Florida ZIP codes and the track of one of the unusual storms

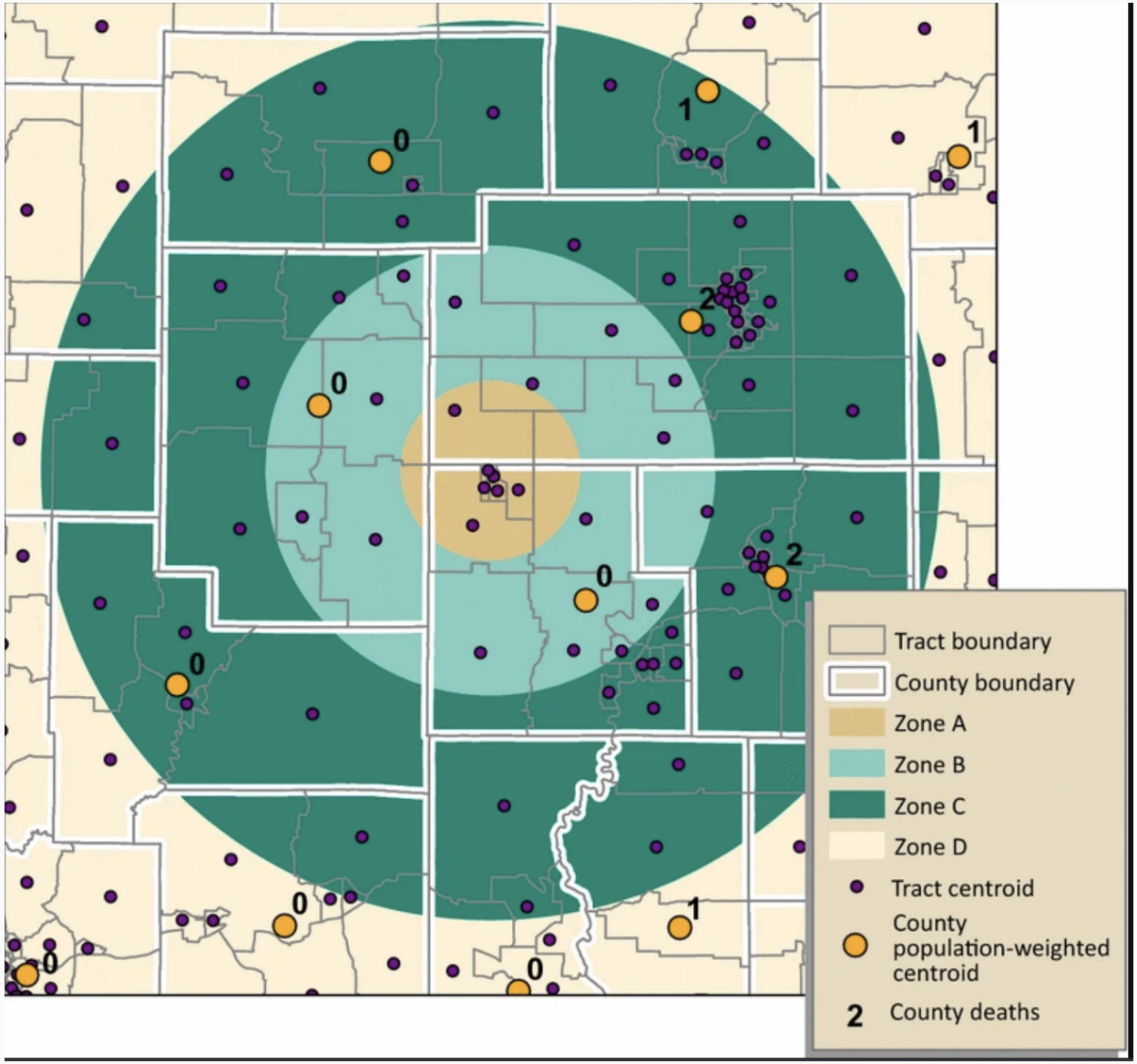
How close have storms come to US counties?

Geographic mean centers



Source: Hallisey et al., 2017

Population mean centers



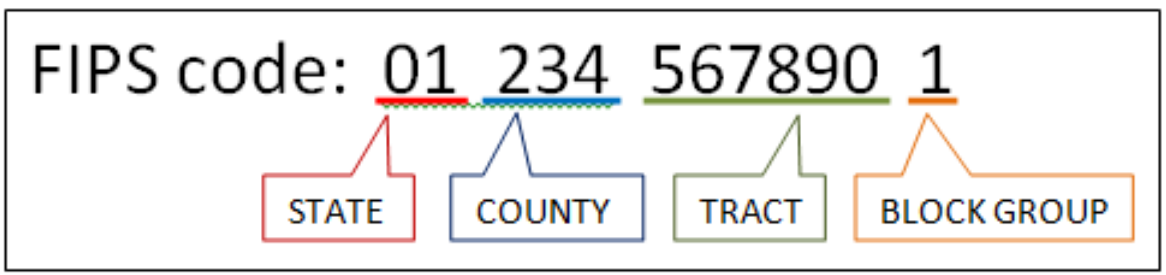
Source: Hallisey et al., 2017

Population mean centers

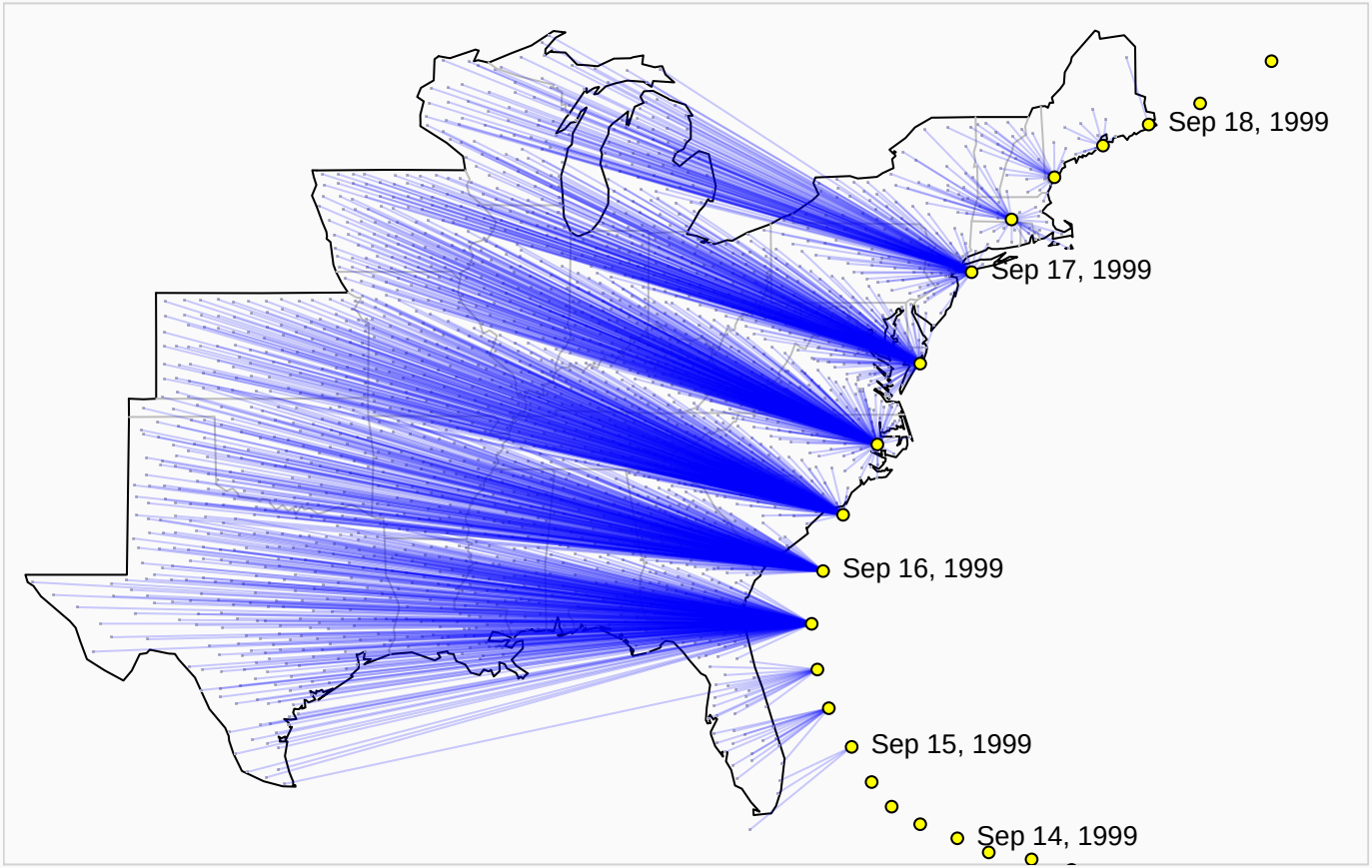
```
data("county_centers")
head(county_centers)
```

```
##    fips county_name state_name population latitude longitude
## 1 01001     Autauga    Alabama      54571 32.50039 -86.49416
## 2 01003     Baldwin    Alabama     182265 30.54892 -87.76238
## 3 01005     Barbour    Alabama      27457 31.84404 -85.31004
## 4 01007        Bibb    Alabama      22915 33.03092 -87.12766
## 5 01009      Blount    Alabama      57322 33.95524 -86.59149
## 6 01011     Bullock    Alabama      10914 32.11633 -85.70119
```

FIPs codes



Finding storm-county minimum distance



Determining all storms that came close to a county

The `county_distance` function from `hurricaneexposure` allows you to determine which of a county were within a certain distance of the storm's track:

```
county_distance(counties = "51700",
                start_year = 1995, end_year = 2005,
                dist_limit = 30)
```

```
##       storm_id  fips closest_date storm_dist       local_time
## 1: Bertha-1996 51700   1996-07-13   28.88430 1996-07-13 04:00
## 2: Gaston-2004 51700   2004-08-30   11.27837 2004-08-30 20:00
##    closest_time_utc
## 1: 1996-07-13 08:00
## 2: 2004-08-31 00:00
```

Determining all storms that came close to a county

The `county_distance` function from `hurricaneexposure` allows you to determine which of a **set** of counties were within a certain distance of the storm's track:

```
county_distance(counties = c("22071", "51700"),
                start_year = 1995, end_year = 2005,
                dist_limit = 30)
```

```
##         storm_id  fips closest_date storm_dist       local_time
## 1:  Bertha-1996 51700   1996-07-13   28.88430 1996-07-13 04:00
## 2:  Bertha-2002 22071   2002-08-05   13.77118 2002-08-05 04:00
## 3: Isidore-2002 22071   2002-09-26    6.37844 2002-09-26 05:45
## 4:  Gaston-2004 51700   2004-08-30   11.27837 2004-08-30 20:00
##    closest_time_utc
## 1: 1996-07-13 08:00
## 2: 2002-08-05 09:00
## 3: 2002-09-26 10:45
## 4: 2004-08-31 00:00
```

Determining all counties close to a storm track

First, use regular expressions to extract all North Carolina county FIPs from the dataset of all counties:

```
nc_counties <- county_centers %>%
   filter(str_sub(fips, 1, 2) == "37") %>%
   pull("fips")
head(nc_counties)
```

```
## [1] "37001" "37003" "37005" "37007" "37009" "37011"
```

Determining all counties close to a storm track

The `county_distance` function from `hurricaneexposure` allows you to determine which of a set of counties were within a certain distance of the storm's track:

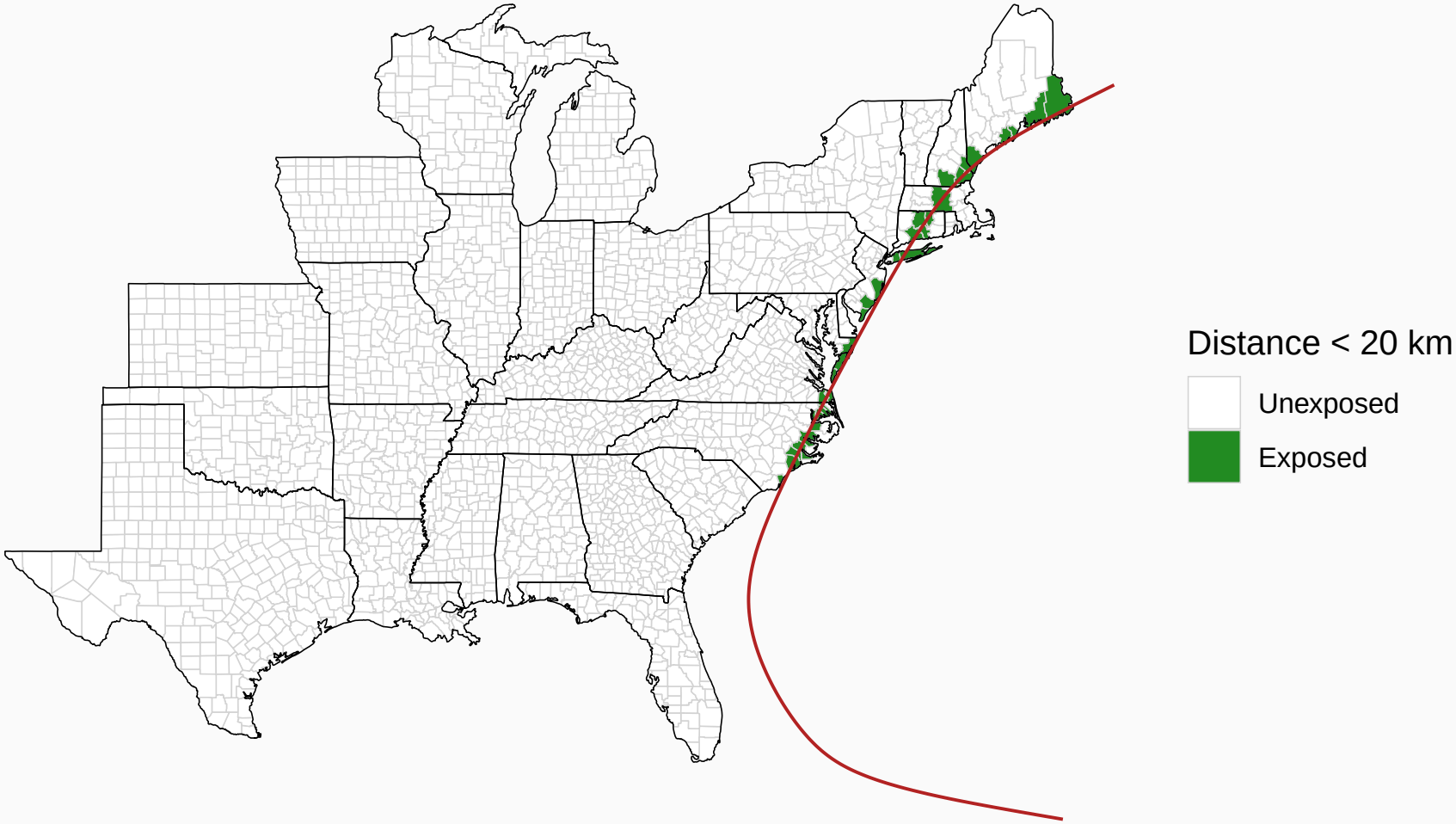
```
county_distance(counties = nc_counties,
                start_year = 1999, end_year = 1999,
                dist_limit = 10) %>%
   filter(storm_id == "Floyd-1999")
```

```
##     storm_id  fips closest_date storm_dist       local_time
## 1 Floyd-1999 37013   1999-09-16   4.591241 1999-09-16 07:30
## 2 Floyd-1999 37041   1999-09-16   8.714198 1999-09-16 09:00
## 3 Floyd-1999 37133   1999-09-16   2.084895 1999-09-16 05:15
## 4 Floyd-1999 37143   1999-09-16   7.276571 1999-09-16 09:30
## 5 Floyd-1999 37187   1999-09-16   3.038735 1999-09-16 08:30
##   closest_time_utc
## 1 1999-09-16 11:30
## 2 1999-09-16 13:00
## 3 1999-09-16 09:15
## 4 1999-09-16 13:30
## 5 1999-09-16 12:30
```

Mapping all counties close to a storm track

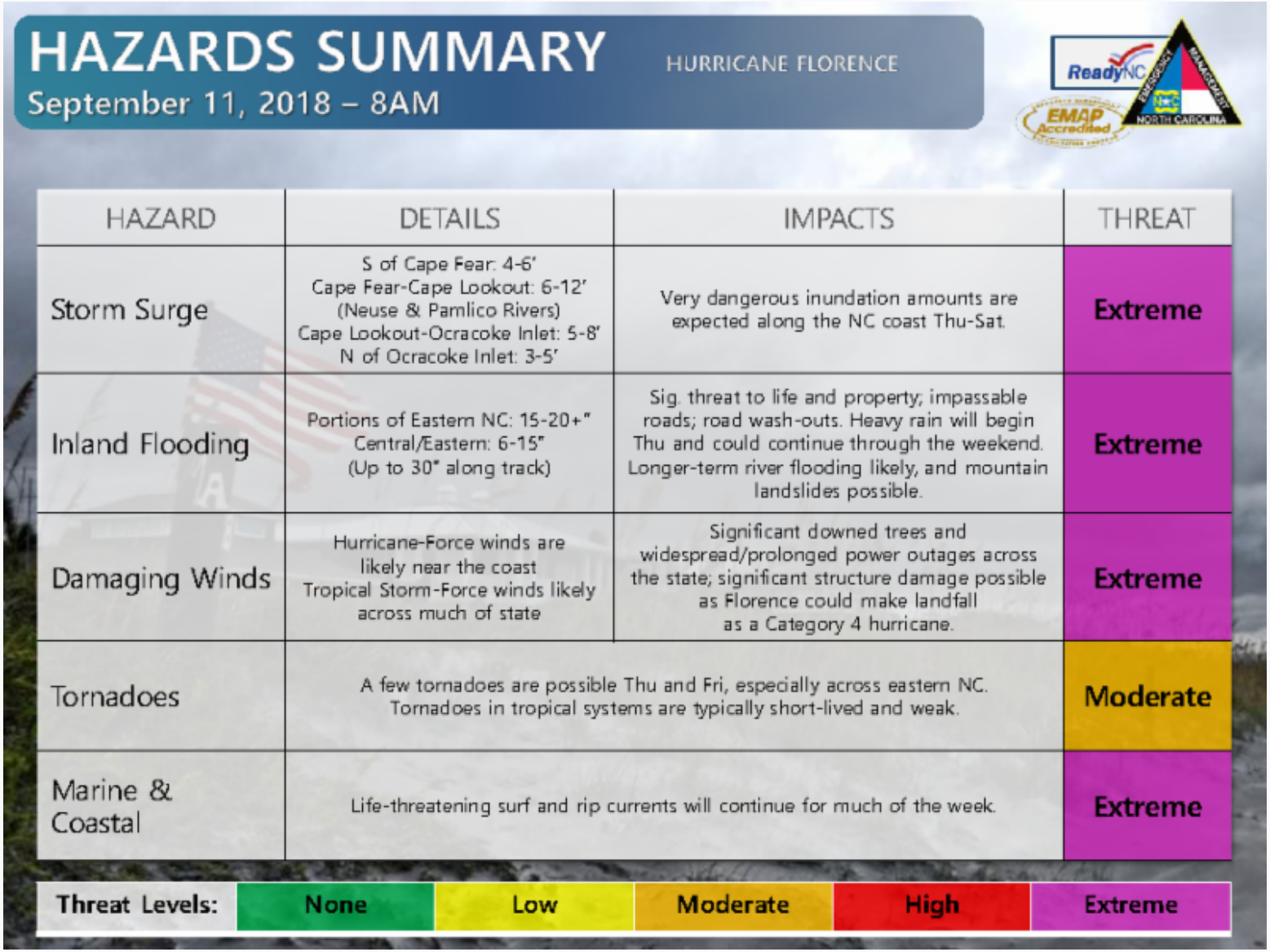
The `map_distance_exposure` function from `hurricaneexposure` allows you to map all counties within a certain distance of a storm's track:

```
map_distance_exposure(storm = "Floyd-1999", dist_limit = 20)
```

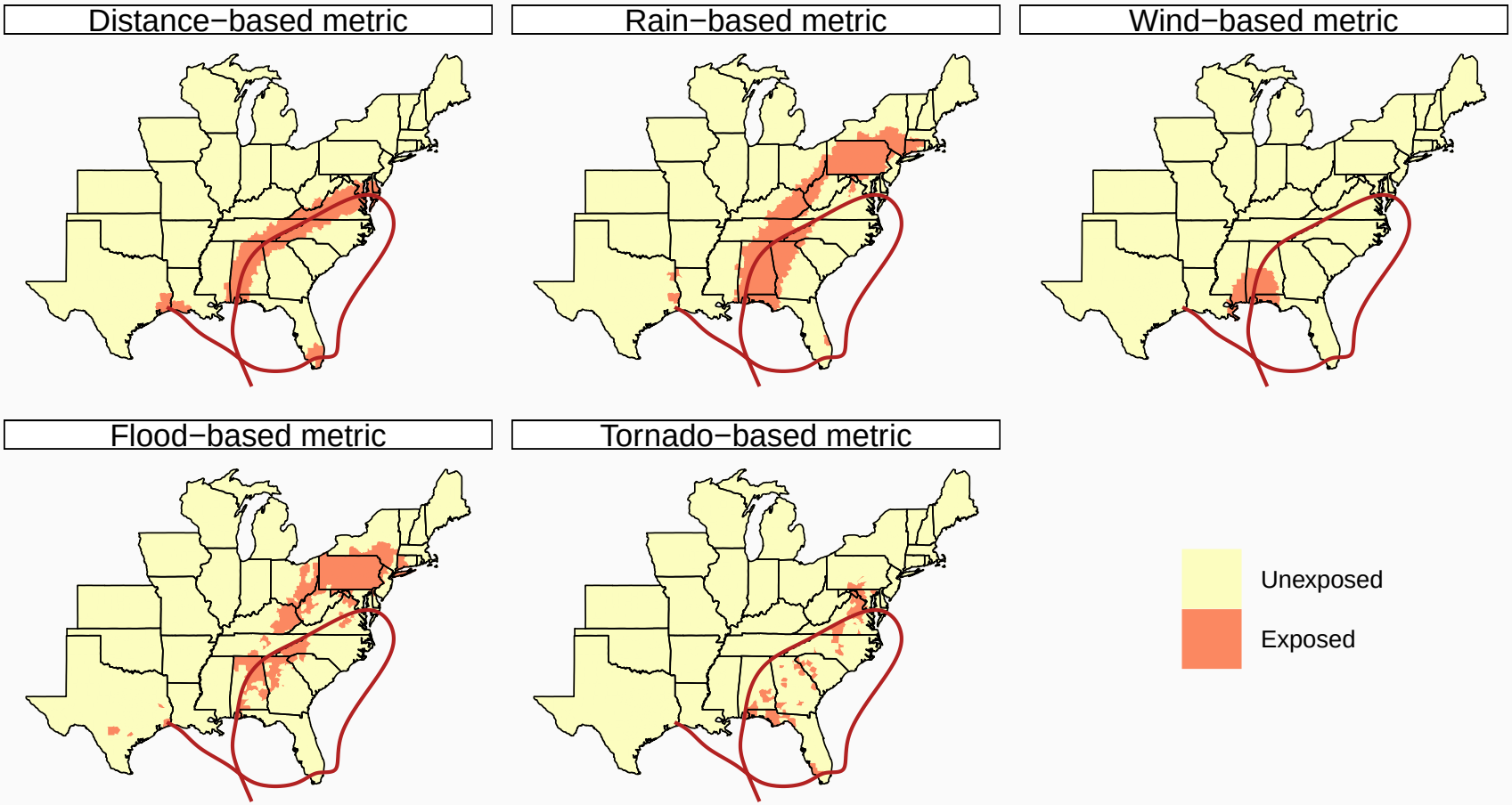


Tropical cyclone hazards

Tropical cyclone hazards

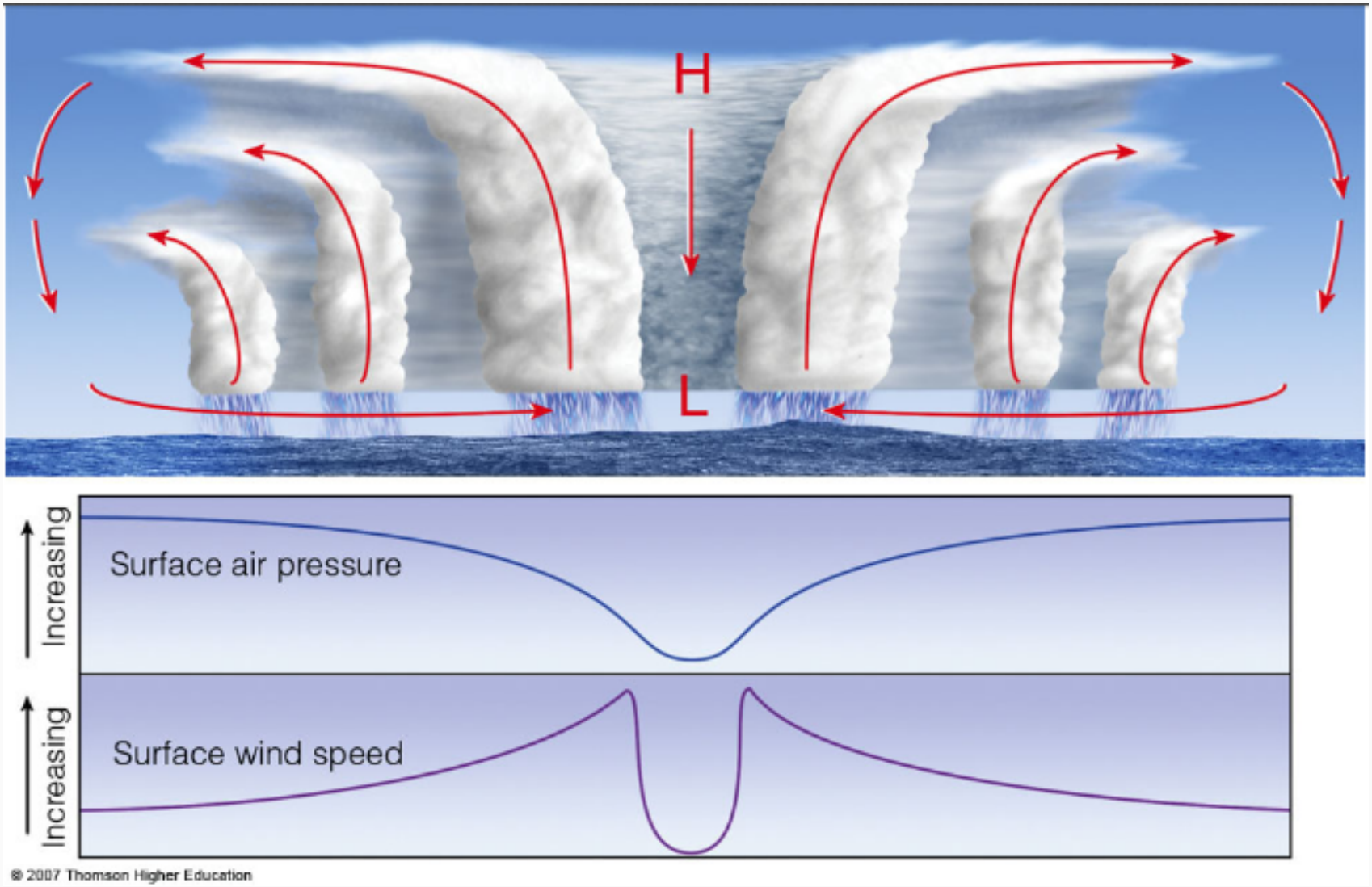


Different patterns in hazards



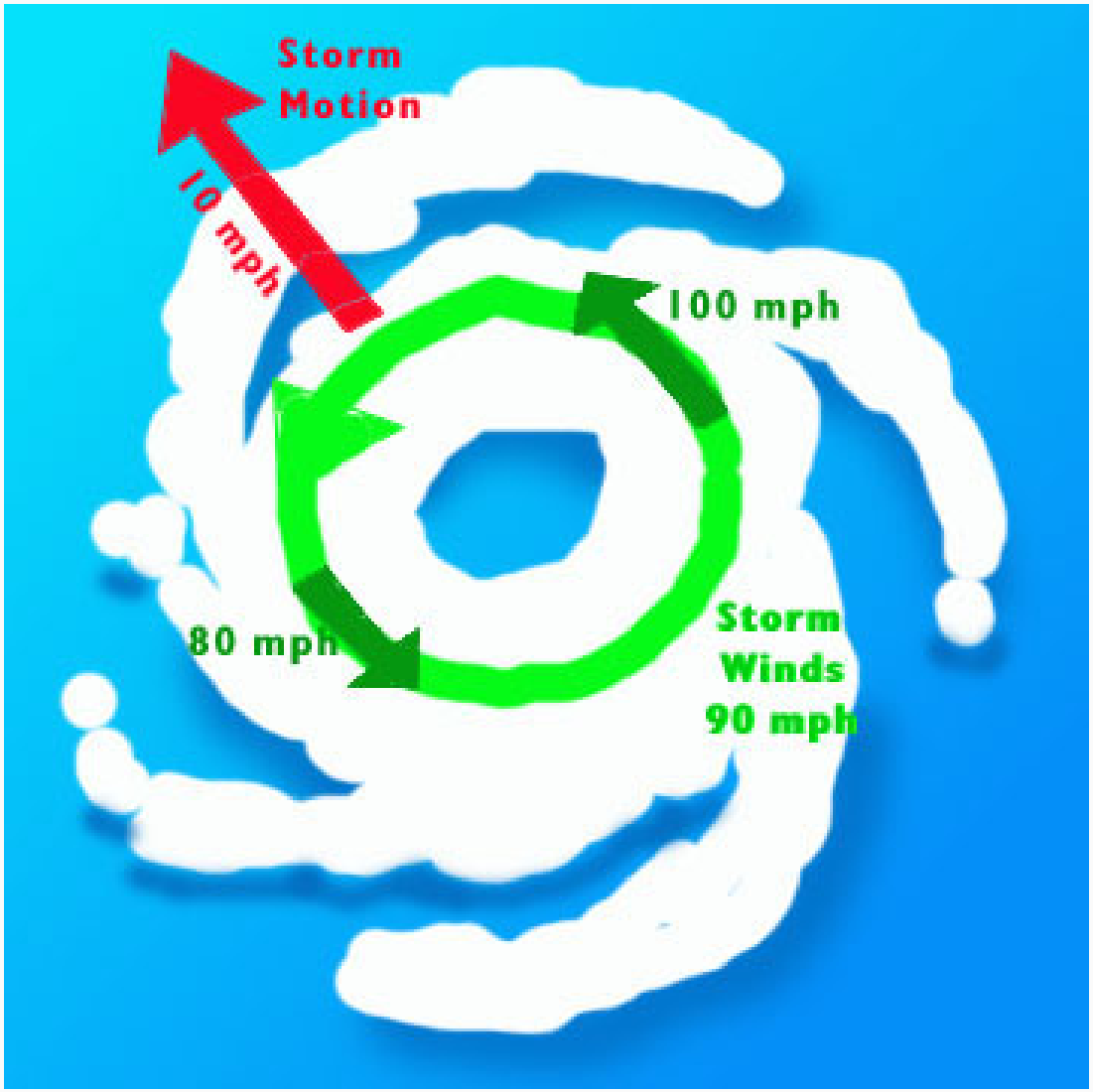
Wind exposure

Modeling storm wind field



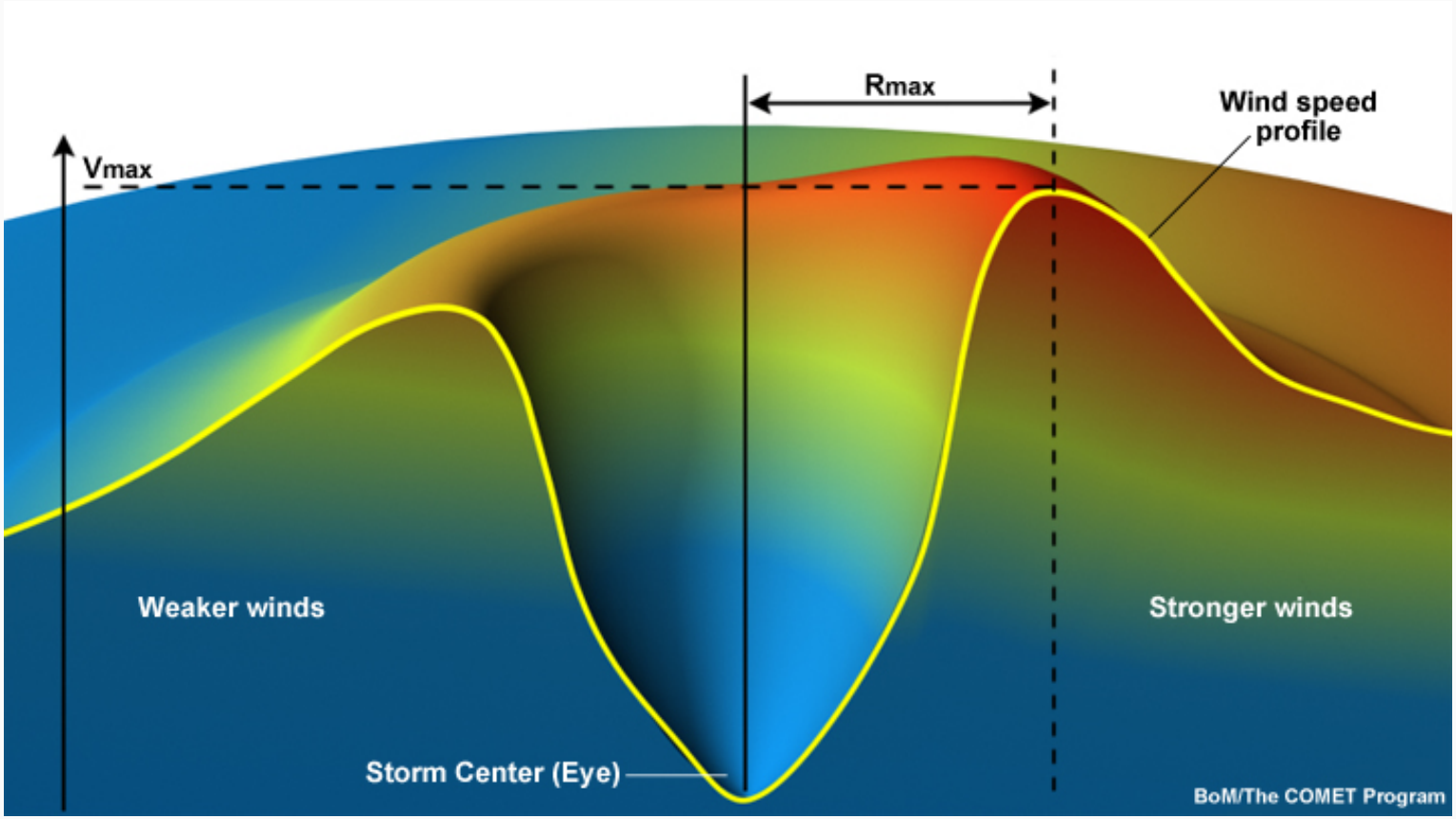
Source: Thompson

Modeling storm wind field



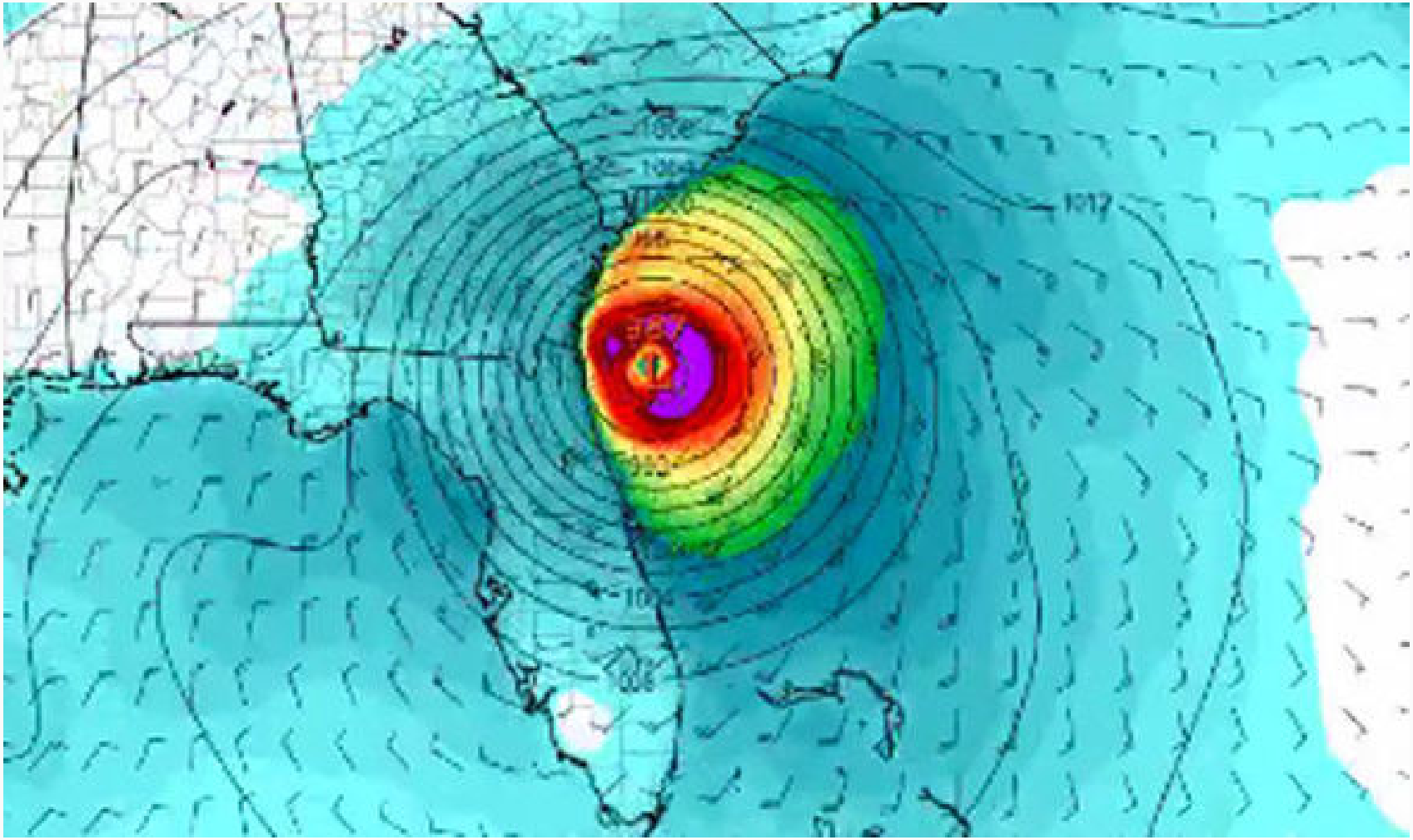
Source: NOAA

Modeling a storm's wind field



Source: NOAA

Modeling storm wind field



Source: ECMWF

Modeling storm wind field

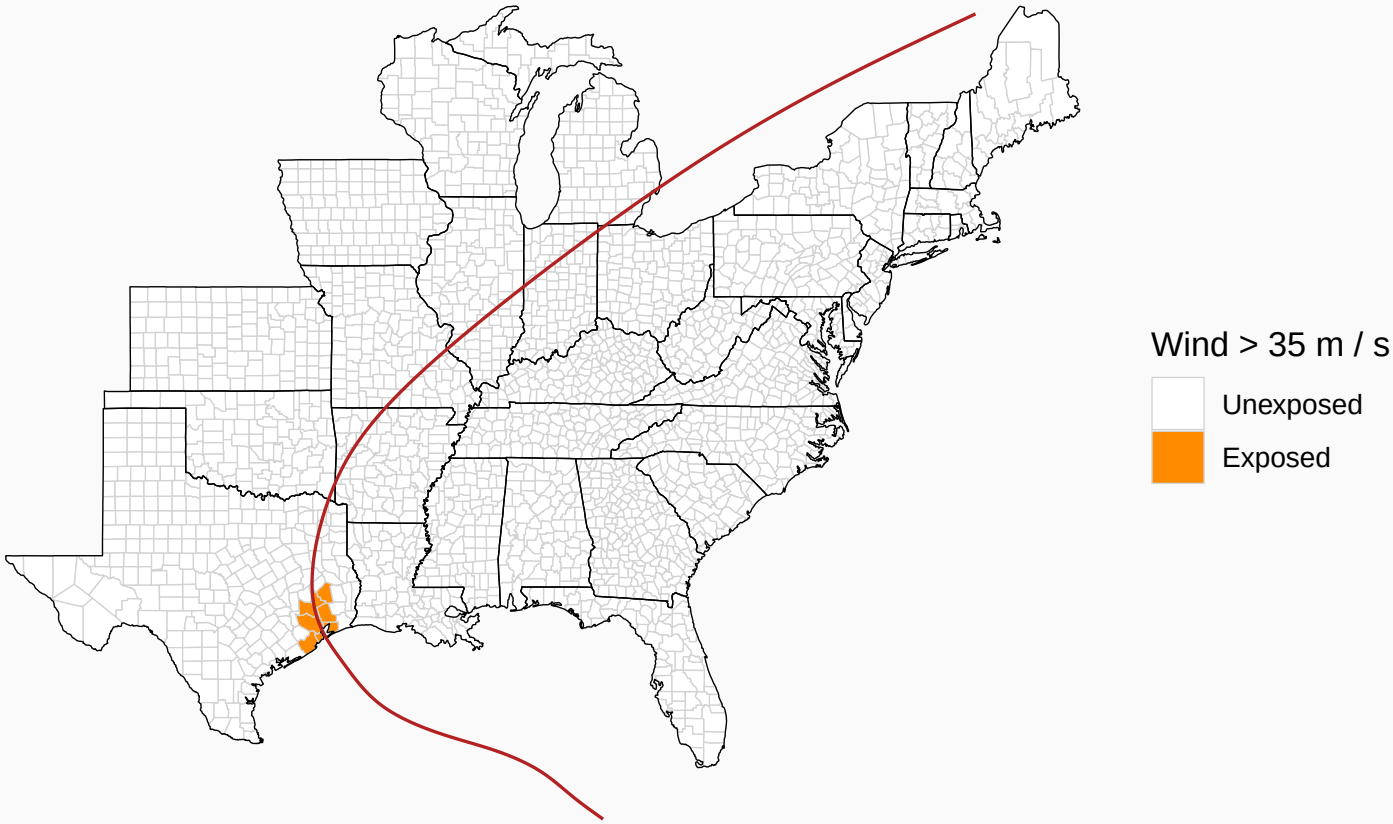
Here is an overview of the wind modeling process implemented by this package:

1. Impute location and maximum wind speed from hurricane track data (every 6 hours) to every 15 minutes.
2. For each storm track location, calculate all the inputs needed for the Willoughby wind speed model (e.g., forward speed, direction of forward motion of the storm).
3. For each county center (or other grid point), estimate surface-level sustained wind every 15 minutes. This step includes: measuring distance to county from storm center (radius); calculating tangential gradient wind components at that grid point; calculating gradient wind direction at that grid point; calculating surface wind speed; calculating surface wind direction, adding storm forward motion back into surface wind estimate.
4. Determine for each county the peak sustained wind throughout the storm.

All wind exposures for a storm

The `map_wind_exposure` function in `hurricaneexposure` allows you to map all counties with peak sustained wind over a certain threshold (in m / s) for a storm:

```
map_wind_exposure(storm = "Ike-2008", wind_limit = 35)
```



All wind exposures for a storm

The `county_wind` function from `hurricaneexposure` allows you to determine which of a set of counties had peak sustained wind above a certain threshold:

```
county_wind(counties = county_centers$fips, wind_limit = 35,
            start_year = 2008, end_year = 2008) %>%
   filter(storm_id == "Ike-2008")
```

```
##   storm_id  fips vmax_sust vmax_gust sust_dur gust_dur closest_time_utc
## 1 Ike-2008 48039  35.18867  52.43112      735     1230 2008-09-13 08:30
## 2 Ike-2008 48071  45.10302  67.20350      840     1290 2008-09-13 09:30
## 3 Ike-2008 48167  43.16761  64.31973      750     1260 2008-09-13 08:00
## 4 Ike-2008 48201  38.67244  57.62194      720     1170 2008-09-13 10:30
## 5 Ike-2008 48291  42.37380  63.13697      765     1230 2008-09-13 11:30
## 6 Ike-2008 48339  36.00104  53.64154      660     1110 2008-09-13 12:15
## 7 Ike-2008 48373  37.51845  55.90249      630     1110 2008-09-13 14:15
## 8 Ike-2008 48407  37.83033  56.36719      600     1125 2008-09-13 13:30
##   storm_dist       local_time closest_date
## 1   48.37654 2008-09-13 03:30   2008-09-13
## 2   29.97233 2008-09-13 04:30   2008-09-13
## 3   11.60959 2008-09-13 03:00   2008-09-13
## 4   31.23934 2008-09-13 05:30   2008-09-13
```

All wind exposures for a county

If you want to get values for a certain region (e.g., a state), start by creating a vector with the county FIPs codes for all counties in that region. For example, to get everything from Florida counties, start by running:

```
fl_counties <- county_centers %>%
   filter(str_sub(fips, 1, 2) == "12") %>%
   pull("fips")
fl_counties %>% head()
```

```
## [1] "12001" "12003" "12005" "12007" "12009" "12011"
```

All wind exposures for a county

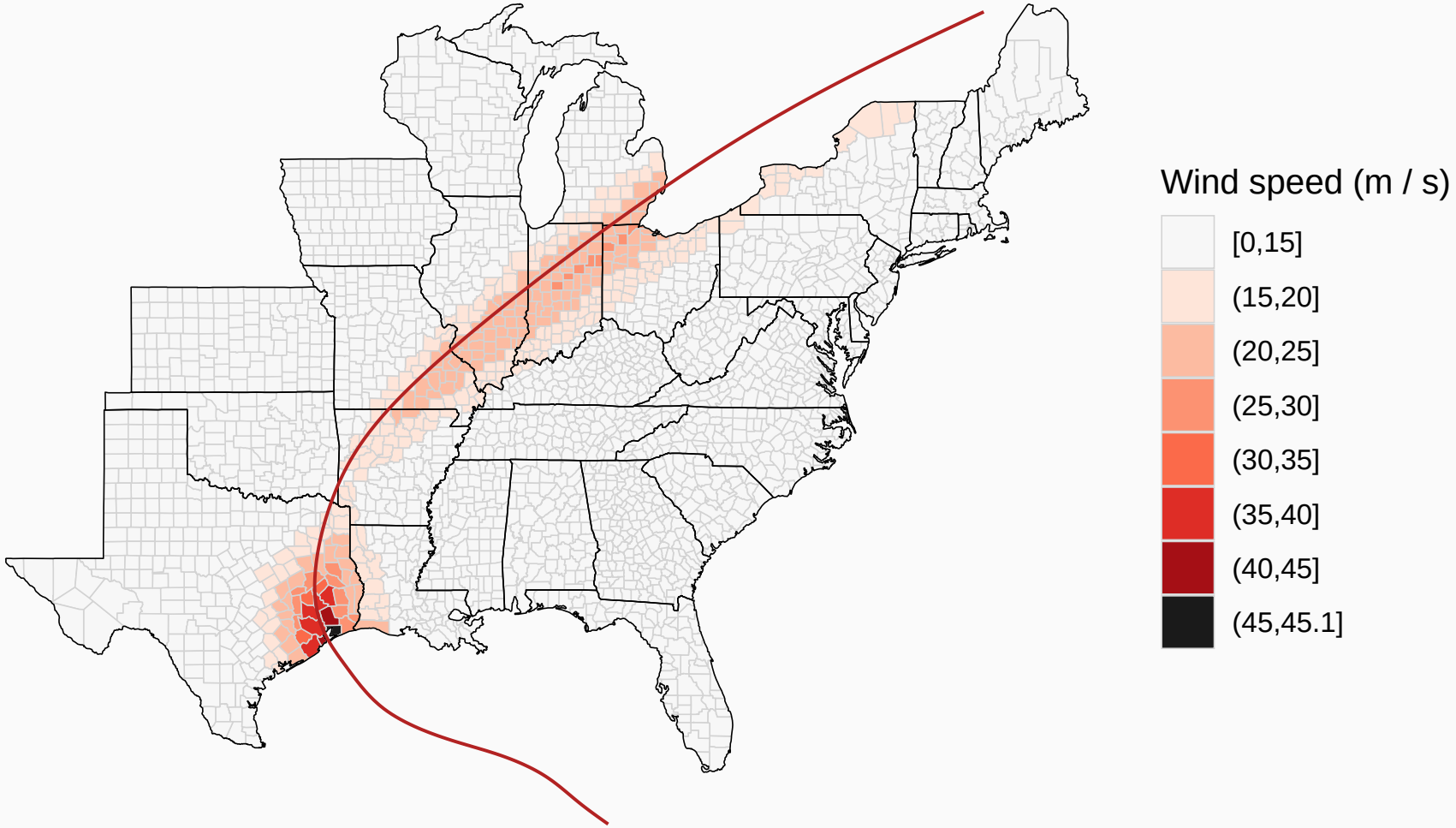
```
county_wind(counties = fl_counties, wind_limit = 45,
            start_year = 2004, end_year = 2004)
```

```
##       storm_id  fips vmax_sust vmax_gust sust_dur gust_dur
## 1 Charley-2004 12015  45.75741  68.17854      450      765
## 2 Charley-2004 12027  46.37071  69.09236      465      765
## 3 Charley-2004 12071  45.29083  67.48334      495      825
## 4 Frances-2004 12111  45.09749  67.19526     1320     2280
## 5  Jeanne-2004 12085  47.17020  70.28360      945     1530
## 6  Jeanne-2004 12111  47.16882  70.28155      885     1500
##   closest_time_utc storm_dist      local_time closest_date
## 1 2004-08-13 20:30   3.369199 2004-08-13 16:30   2004-08-13
## 2 2004-08-13 21:15  12.867691 2004-08-13 17:15   2004-08-13
## 3 2004-08-13 19:45  35.198896 2004-08-13 15:45   2004-08-13
## 4 2004-09-05 08:45   7.833093 2004-09-05 04:45   2004-09-05
## 5 2004-09-26 04:00  13.141154 2004-09-26 00:00   2004-09-26
## 6 2004-09-26 04:45   5.680629 2004-09-26 00:45   2004-09-26
```

All wind exposures for a storm

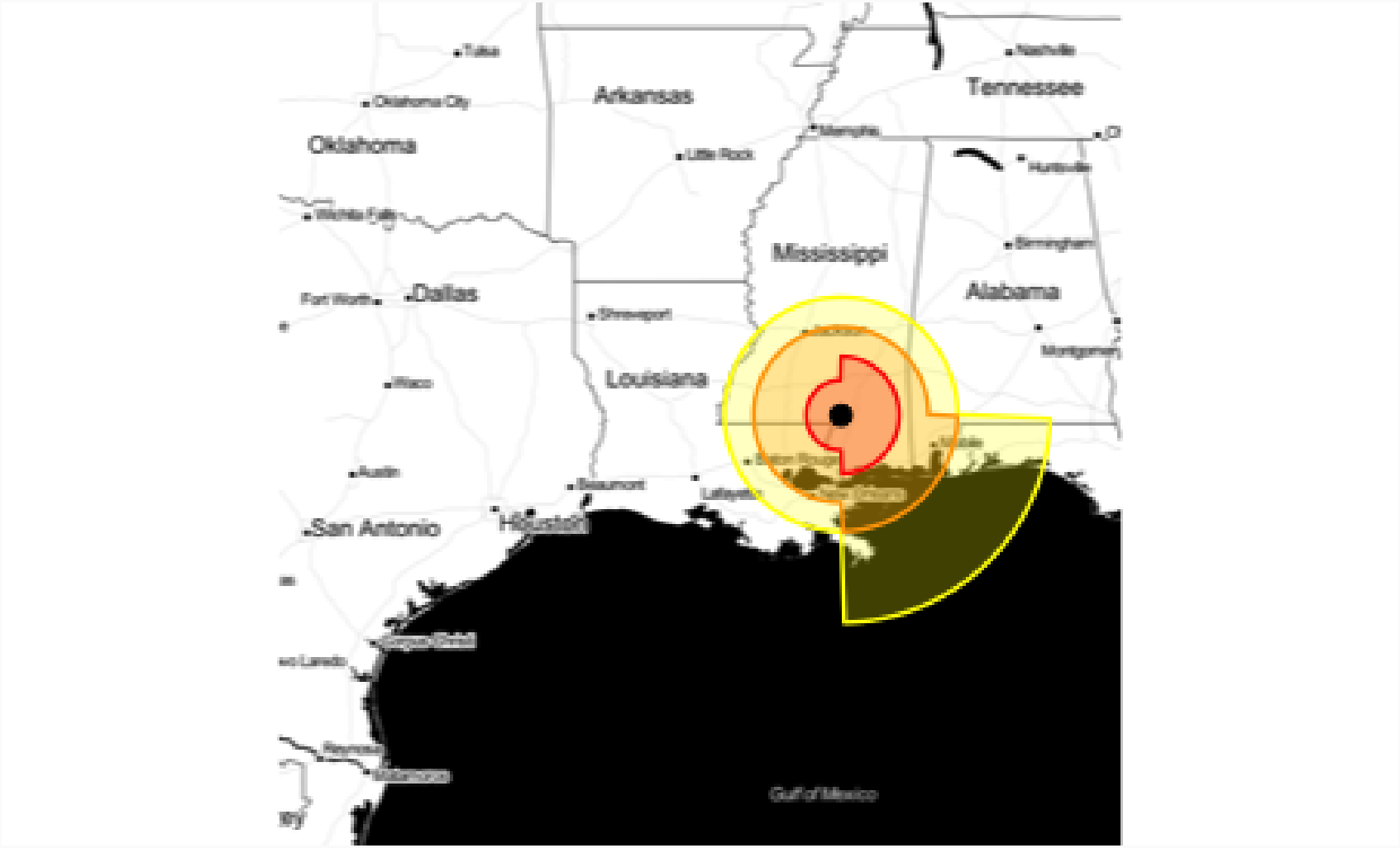
Use `map_counties` with `metric = "wind"` to map county-level peak sustained wind from a storm:

```
map_counties(storm = "Ike-2008", metric = "wind")
```



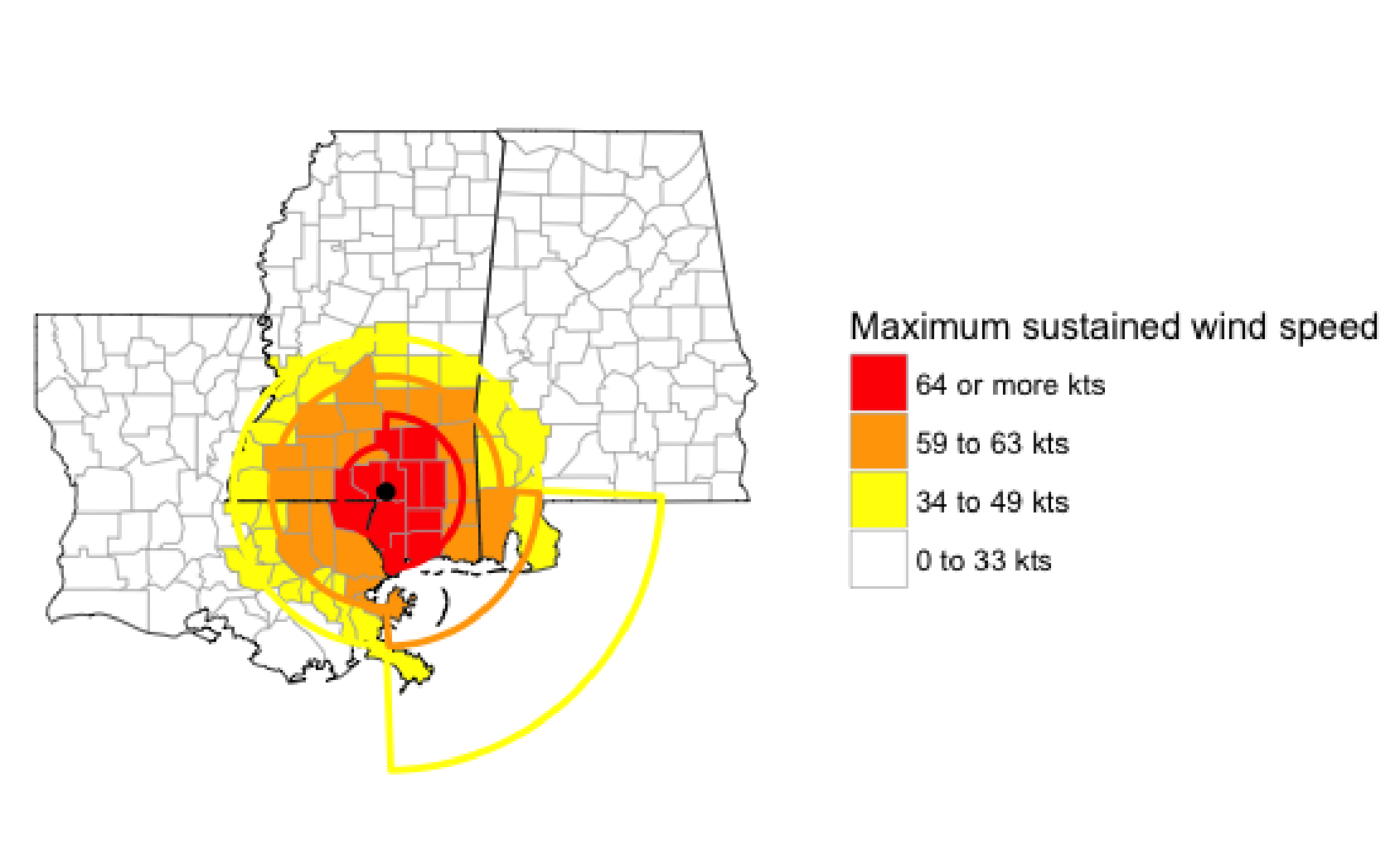
Other sources of wind exposure

Wind radii: estimates of extent of peak sustained winds of certain speeds.

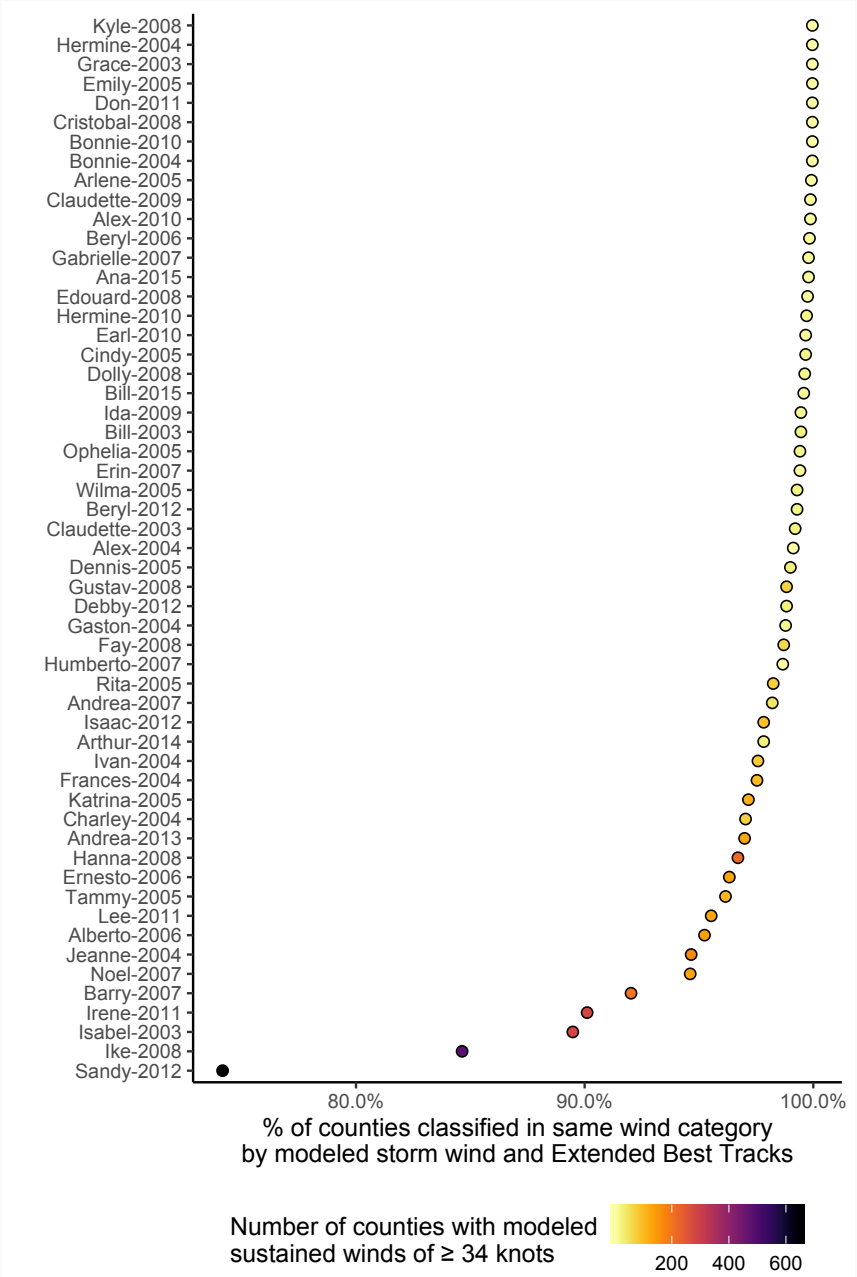


Other sources of wind exposure

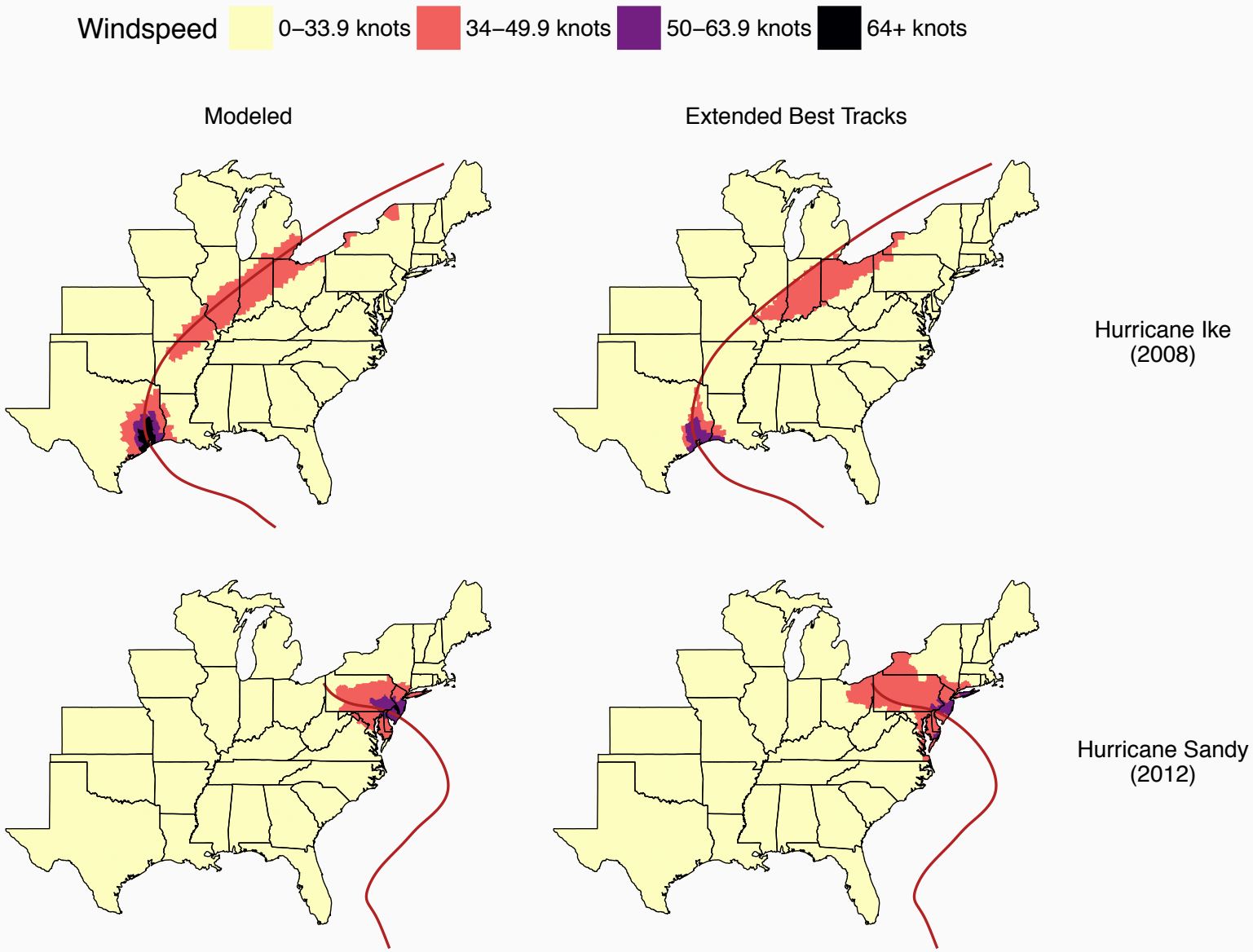
Wind radii: estimates of extent of peak sustained winds of certain speeds.



Comparison of wind data



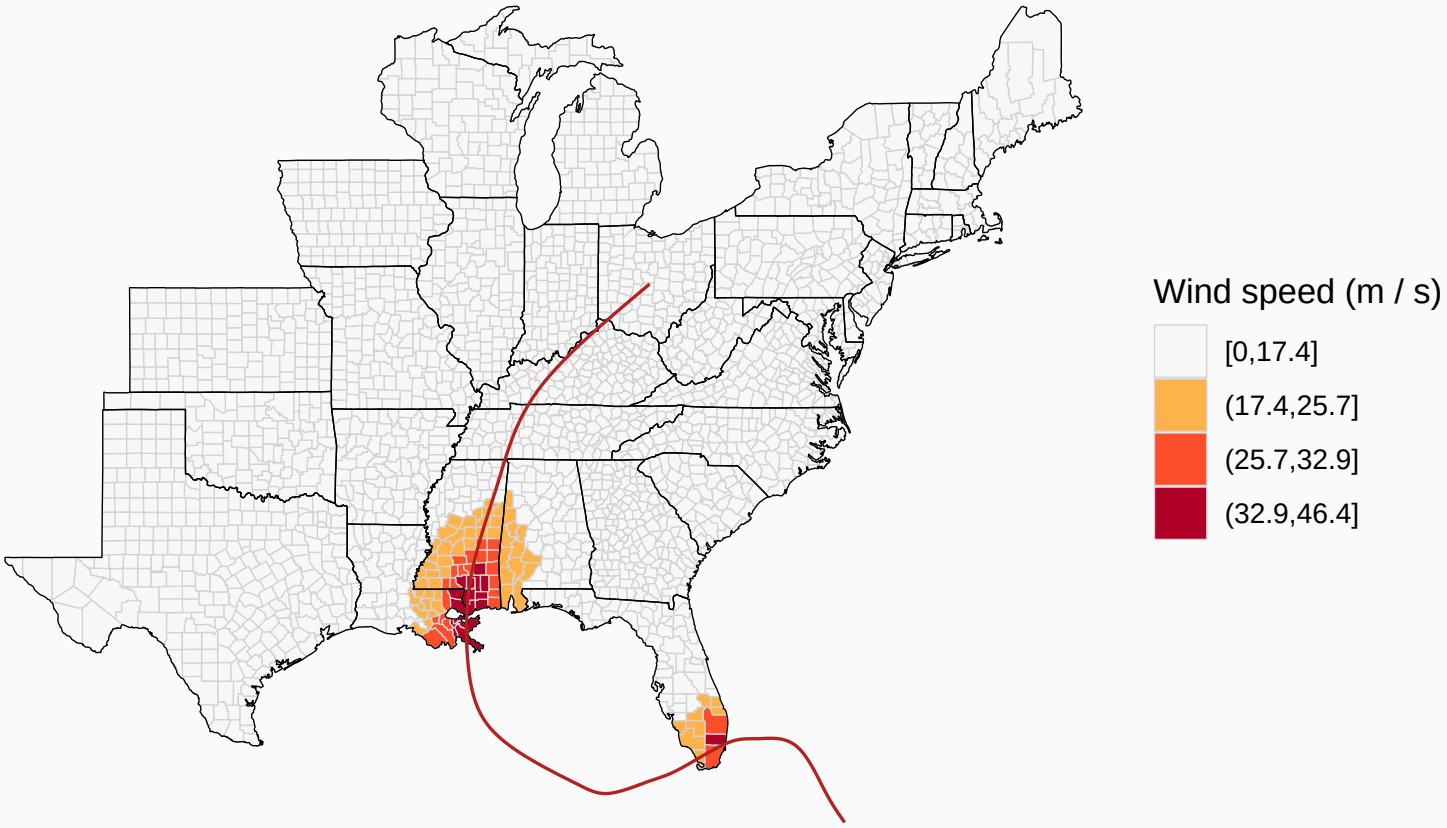
Comparison of wind data



Using wind radii–based metrics

You can change to use the wind radii (instead of the modeled wind) by using the "ext_tracks" option for the `wind_source` parameter (available since 2004):

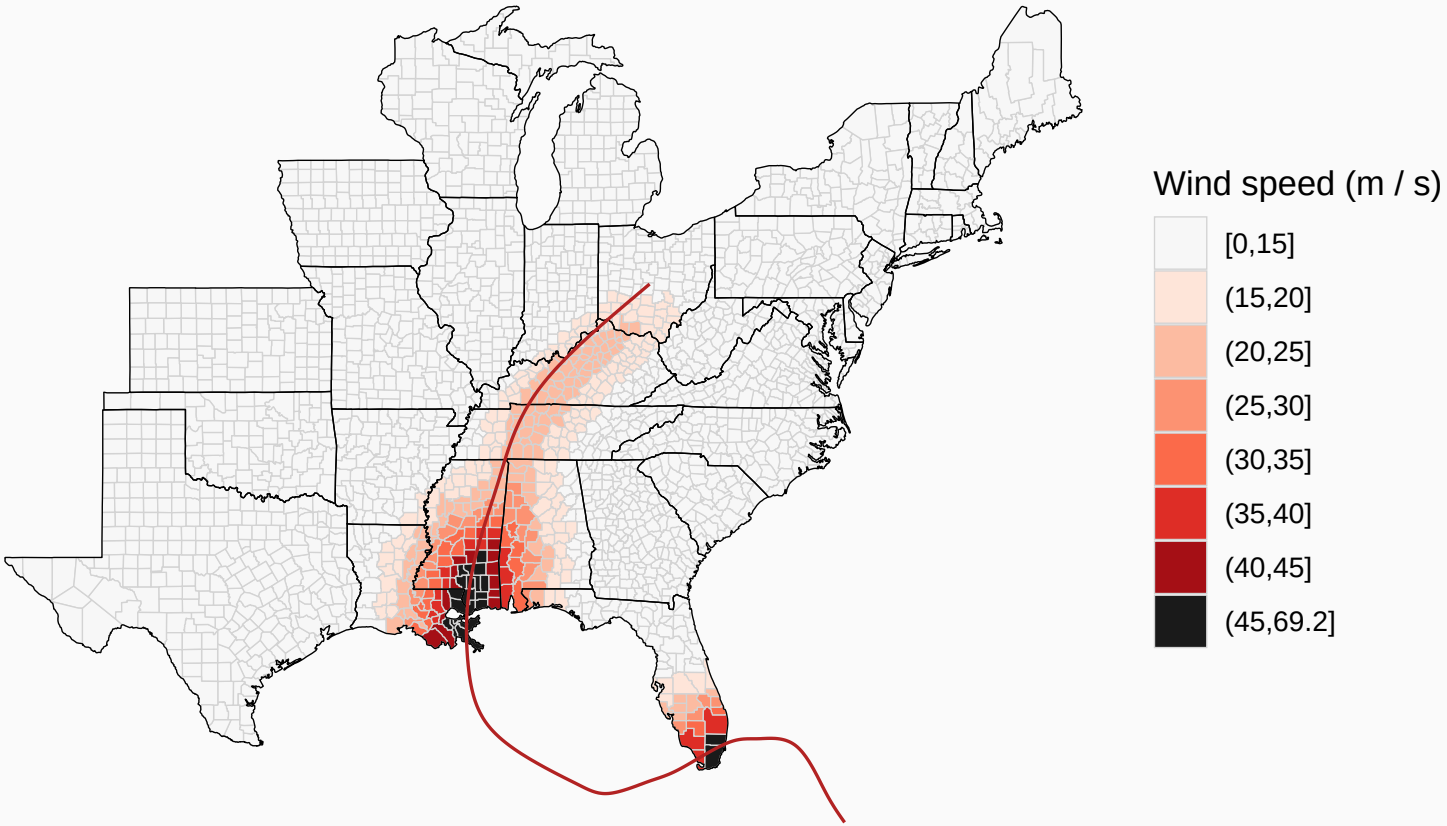
```
map_counties(storm = "Katrina-2005", metric = "wind",
             wind_source = "ext_tracks")
```



Other wind metrics: Gust

You can use the "vmax_gust" option to the `wind_var` parameter to use gust wind estimates unstead of sustained wind estimates (note: this uses a constant gust factor for conversion):

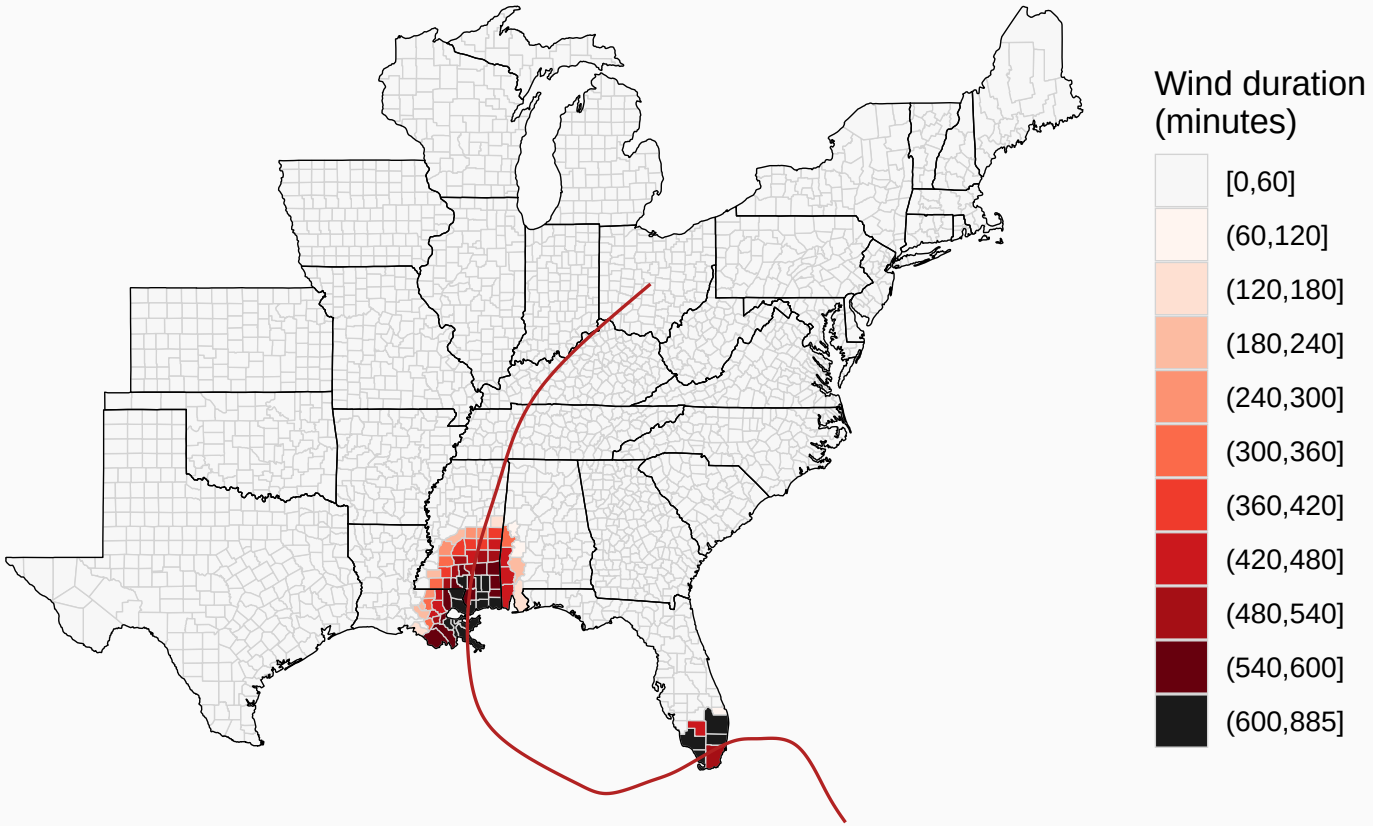
```
map_counties(storm = "Katrina-2005", metric = "wind",
             wind_var = "vmax_gust")
```



Other wind metrics: Duration

You can use the "sust_dur" option to the `wind_var` parameter to use gust wind estimates unstead of sustained wind estimates (note: this uses a constant gust factor for conversion):

```
map_counties(storm = "Katrina-2005", metric = "wind",
             wind_var = "sust_dur")
```



Wind data

You can also access the dataframe with all tropical cyclone wind data, using `storm_winds`:

```
data("storm_winds")
head(storm_winds)
```

```
##    fips vmax_gust vmax_sust gust_dur sust_dur     storm_id
## 1 01001  1.168851 0.7844640        0        0 Alberto-1988
## 2 01003  1.000133 0.6712300        0        0 Alberto-1988
## 3 01005  1.522113 1.0215523        0        0 Alberto-1988
## 4 01007  1.010524 0.6782040        0        0 Alberto-1988
## 5 01009  1.007067 0.6758839        0        0 Alberto-1988
## 6 01011  1.387692 0.9313369        0        0 Alberto-1988
```

Wind data—wind radii–based estimates

The wind radii–based estimates are available in the `ext_tracks_wind` data:

```
data("ext_tracks_wind")
head(ext_tracks_wind)
```

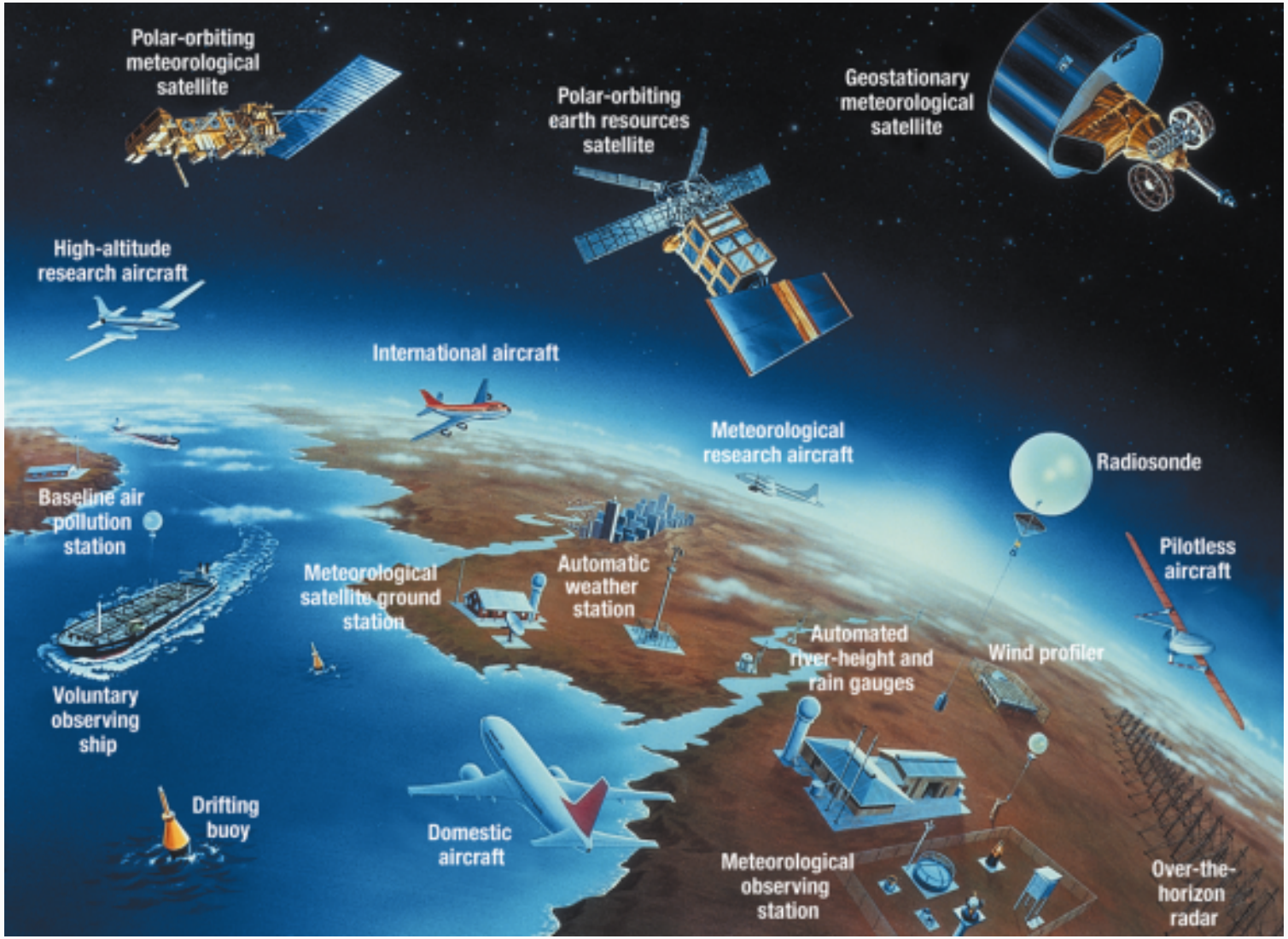
```
## # A tibble: 6 x 5
##   fips  vmax_gust vmax_sust sust_dur storm_id
##   <chr>     <dbl>     <dbl>    <int> <chr>
## 1 01001         0         0        0 Alex-2004
## 2 01003         0         0        0 Alex-2004
## 3 01005         0         0        0 Alex-2004
## 4 01007         0         0        0 Alex-2004
## 5 01009         0         0        0 Alex-2004
## 6 01011         0         0        0 Alex-2004
```

Your turn

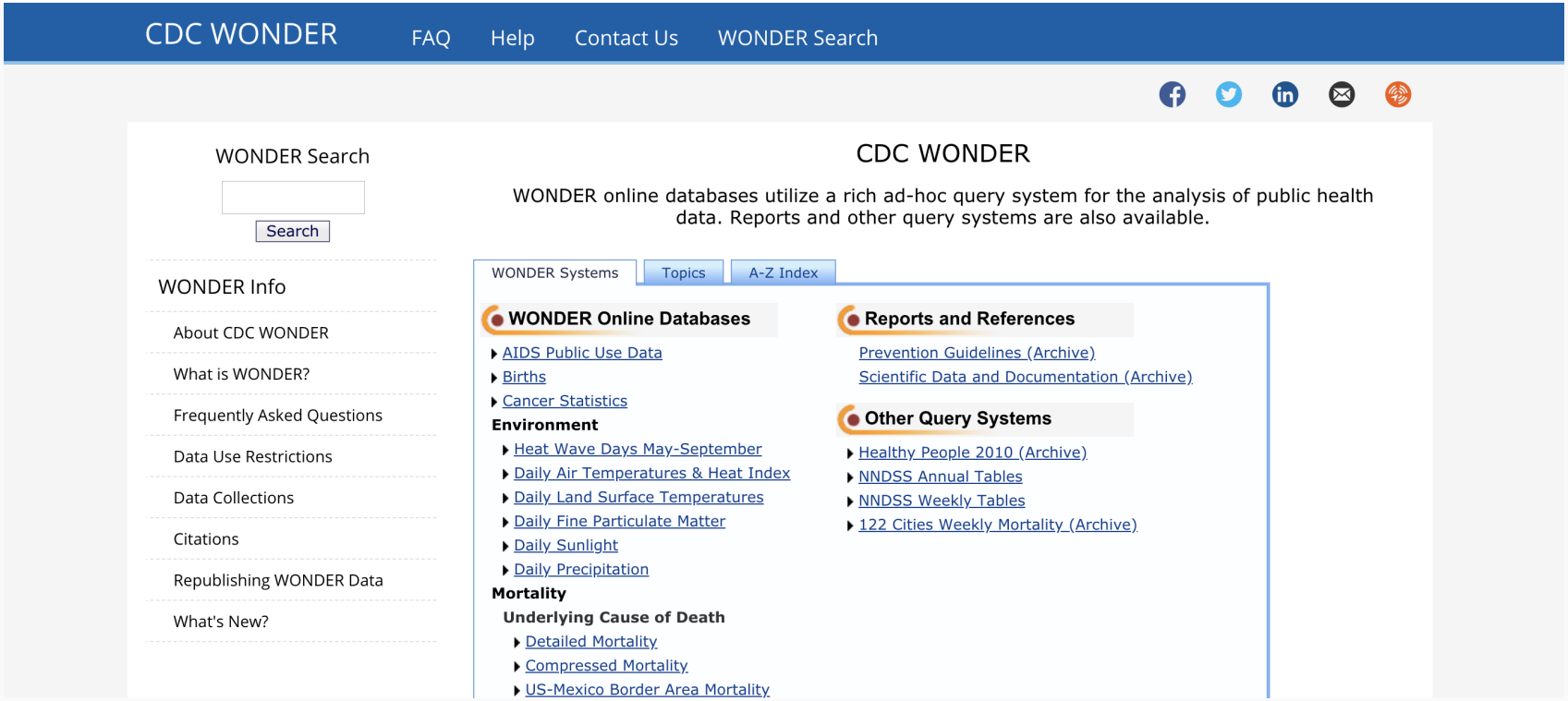
- Determine all the storms that brought winds of 17.5 m / s or higher to Miami-Dade County, FL between 1990 and 2015
- Which brought the most intense wind to Miami-Dade County? Create a map showing the peak sustained wind that storm brought to each eastern US county.
- Create the same map, but use the wind radii–based wind exposure values rather than the modeled wind values.

Rain exposure

Re-analysis weather data

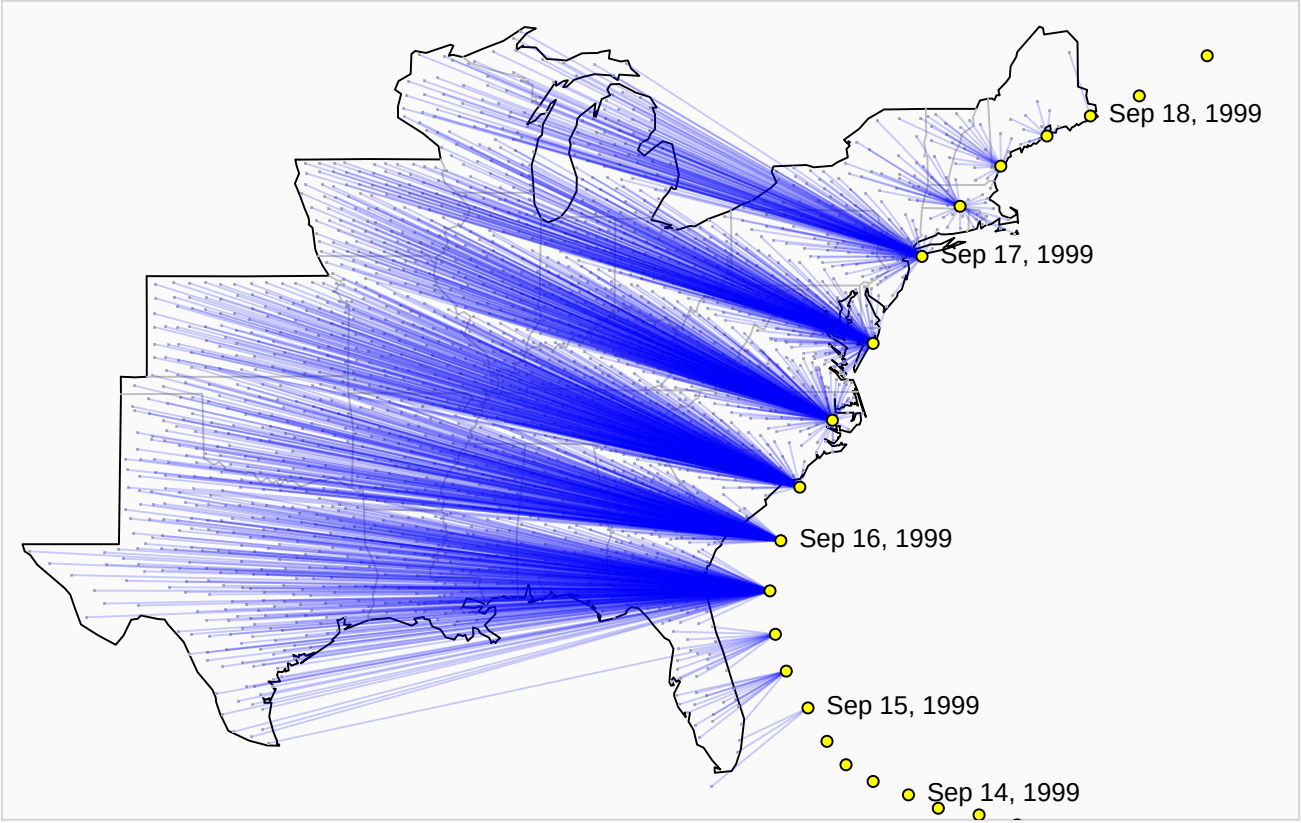


CDC WONDER database

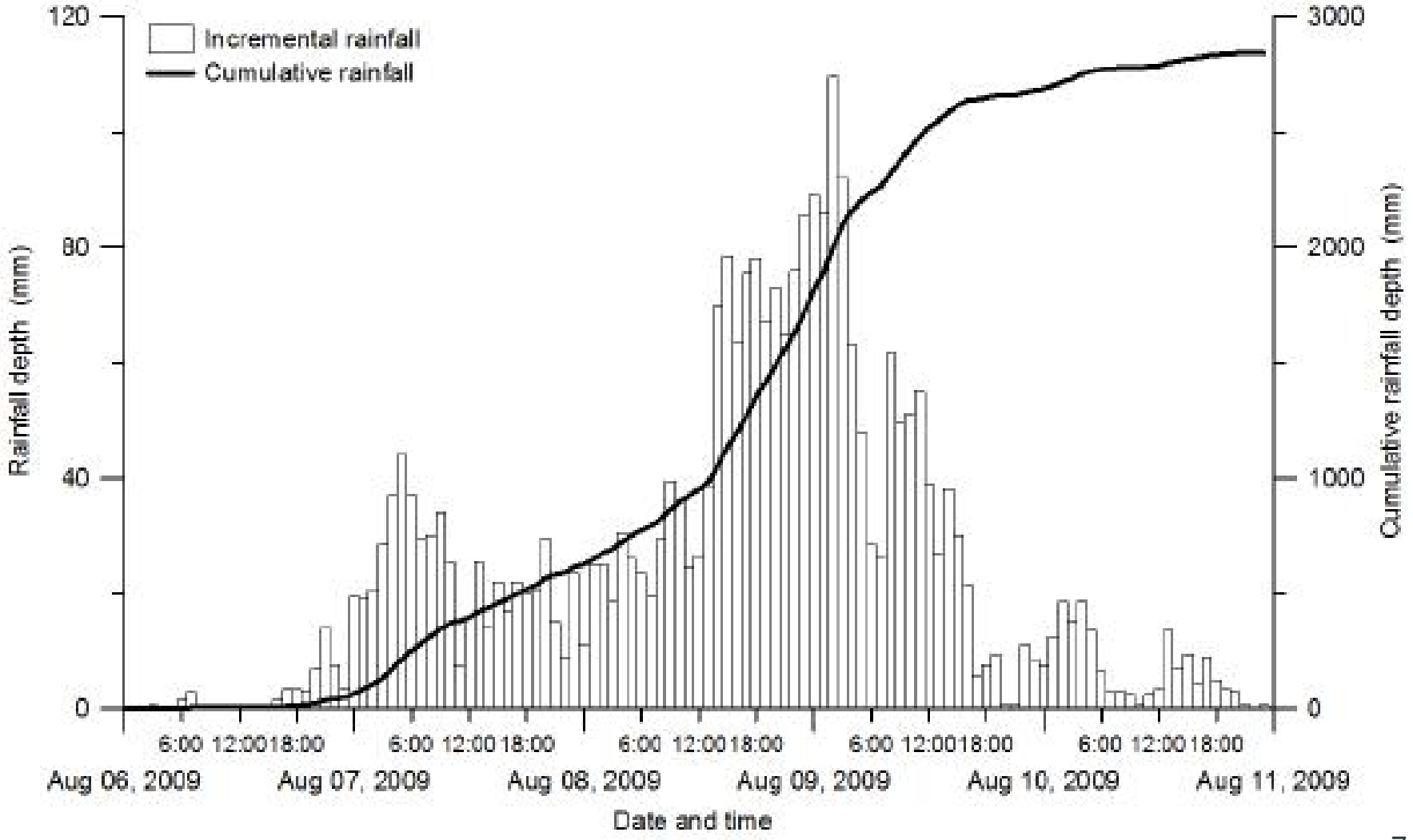


Source: US Centers for Disease Control and Prevention

Matching with storm tracks



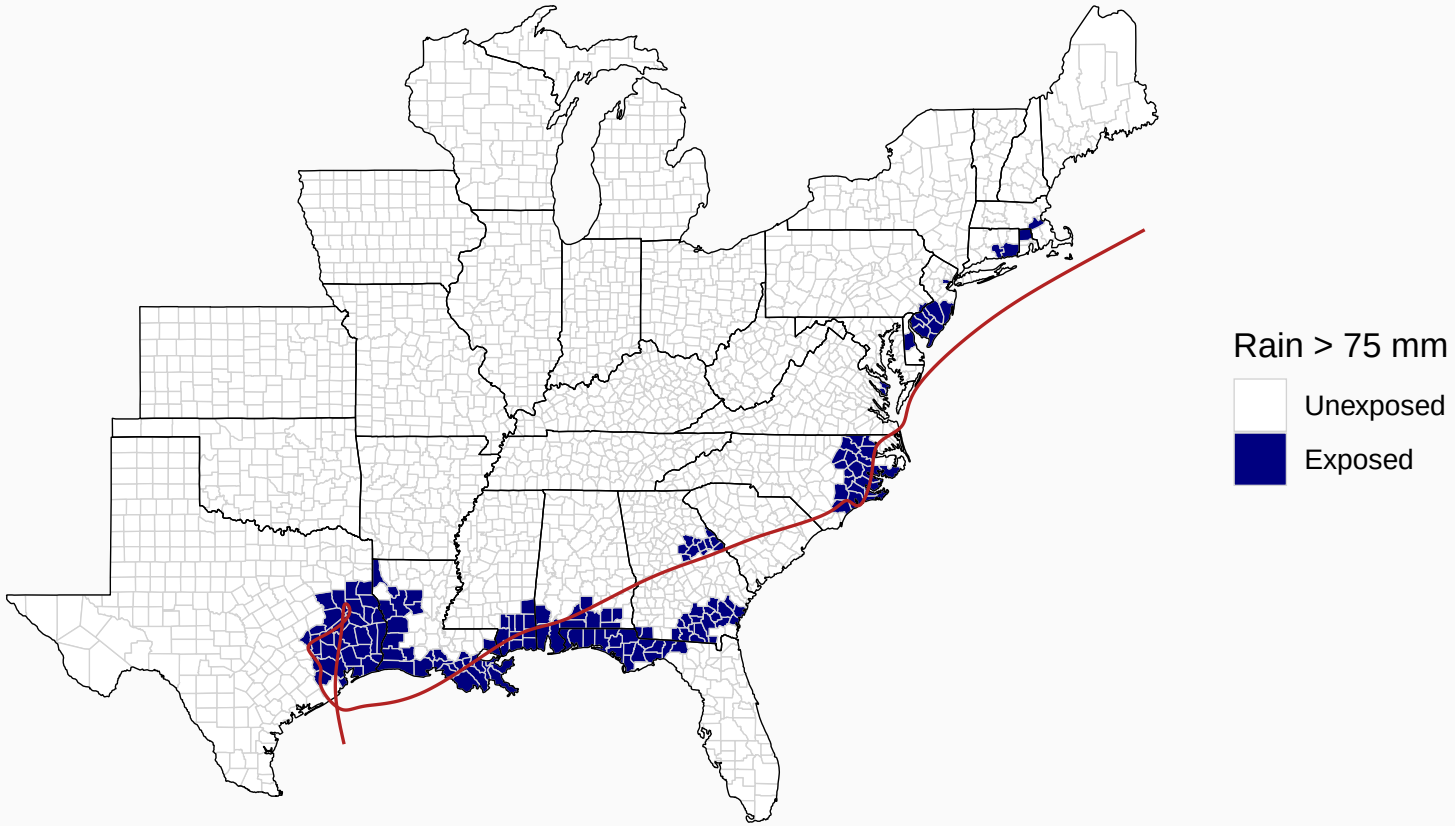
Cumulative rainfall



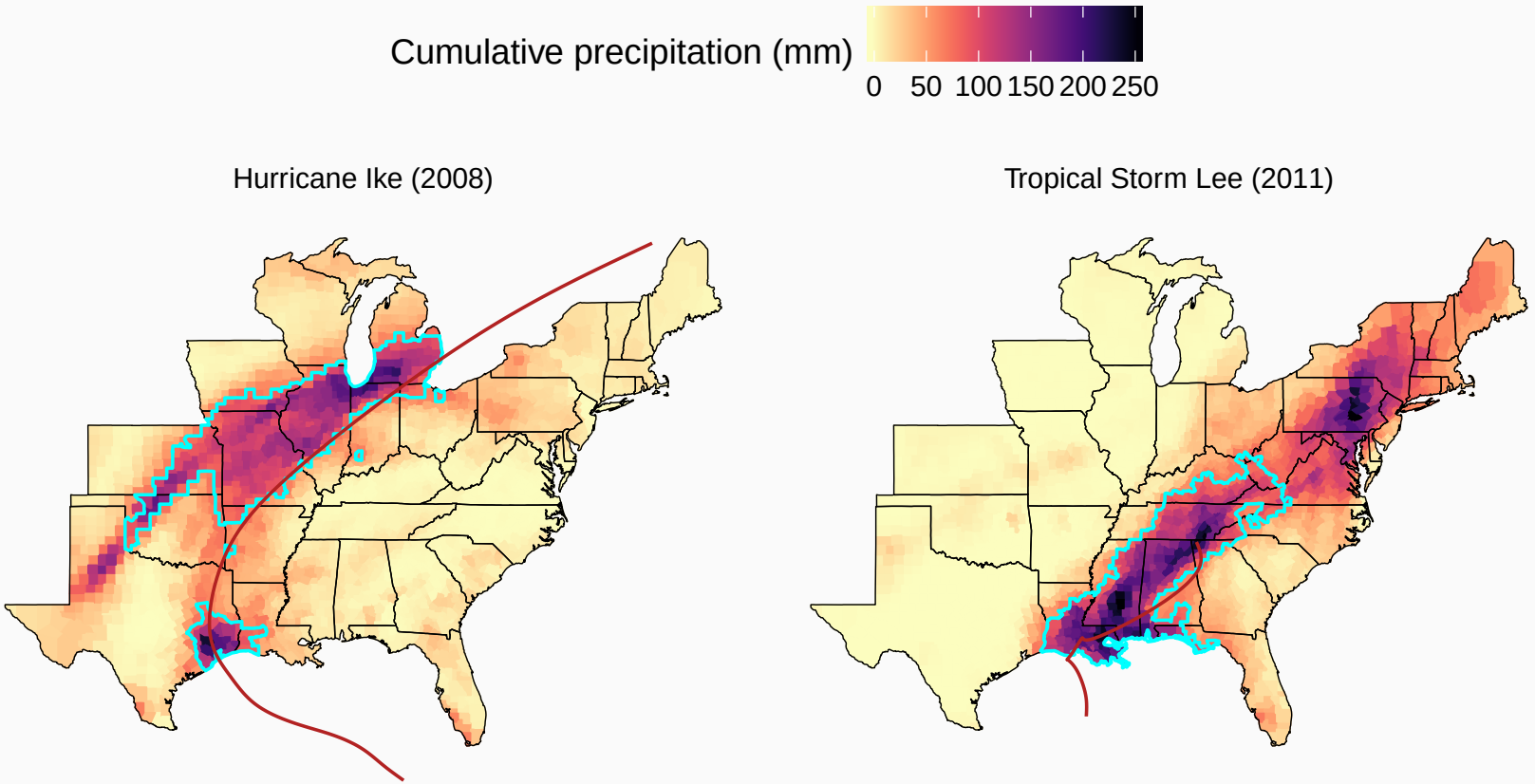
All rain exposures for a storm

The `map_rain_exposure` function in `hurricaneexposure` allows you to map all counties with cumulative precipitation over a certain threshold (in mm) for a storm:

```
map_rain_exposure(storm = "Allison-2001", rain_limit = 75,
                  dist_limit = 250)
```



Distance constraint



All rain exposures for a storm

The `county_rain` function from `hurricaneexposure` allows you to determine which of a set of counties had cumulative rainfall above a certain threshold:

```
all_fips <- county_centers %>% pull("fips")
county_rain(counties = all_fips,
            rain_limit = 150, dist_limit = 250,
            start_year = 2001, end_year = 2001) %>%
   filter(storm_id == "Allison-2001")
```

```
##       storm_id  fips closest_date storm_dist tot_precip       local_
## 1 Allison-2001 22057   2001-06-11  28.350391      152.4 2001-06-11 0
## 2 Allison-2001 48005   2001-06-07  17.287200      170.4 2001-06-07 0
## 3 Allison-2001 48185   2001-06-08  21.118270      179.9 2001-06-08 0
## 4 Allison-2001 48455   2001-06-07   5.013884      169.8 2001-06-07 1
## 5 Allison-2001 48471   2001-06-07  15.764435      154.2 2001-06-07 1
##   closest_time_utc
## 1 2001-06-11 05:45
## 2 2001-06-07 13:00
## 3 2001-06-08 07:45
```

All rain exposures for a county

If you want to get values for a certain region (e.g., a state), start by creating a vector with the county FIPs codes for all counties in that region. For example, to get everything from Texas counties, start by running:

```
tx_counties <- county_centers %>%
   filter(str_sub(fips, 1, 2) == "48") %>%
   pull("fips")
tx_counties %>% head()
```

```
## [1] "48001" "48003" "48005" "48007" "48009" "48011"
```

All rain exposures for a county

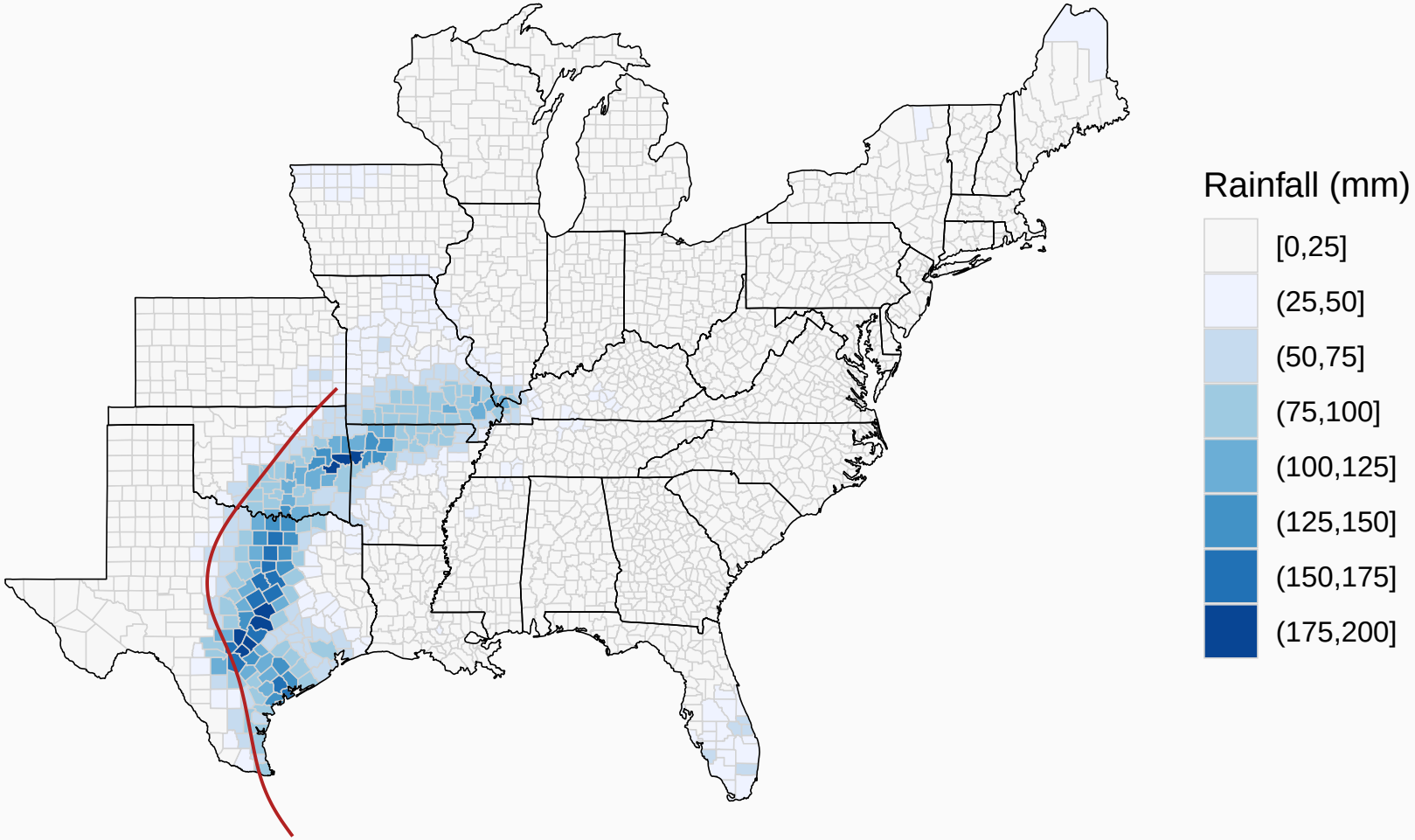
```
county_rain(counties = tx_counties, rain_limit = 175,
            dist_limit = 250,
            start_year = 2009, end_year = 2011)
```

```
##         storm_id  fips closest_date storm_dist tot_precip       local
## 1:    Alex-2010 48215   2010-06-30  210.74322      183.3 2010-06-30
## 2: Hermine-2010 48027   2010-09-07  159.62482      181.4 2010-09-07
## 3: Hermine-2010 48031   2010-09-07   51.76863      175.5 2010-09-07
## 4: Hermine-2010 48091   2010-09-07   49.69290      187.8 2010-09-07
## 5: Hermine-2010 48209   2010-09-07   85.01687      186.1 2010-09-07
## 6: Hermine-2010 48491   2010-09-07  127.78742      187.4 2010-09-07
##    closest_time_utc
## 1: 2010-07-01 01:45
## 2: 2010-09-08 00:00
## 3: 2010-09-07 21:00
## 4: 2010-09-07 19:15
## 5: 2010-09-07 19:45
## 6: 2010-09-07 21:45
```

All rain exposures for a county

Use `map_counties` with `metric = "rain"` to map county-level peak sustained wind from a storm:

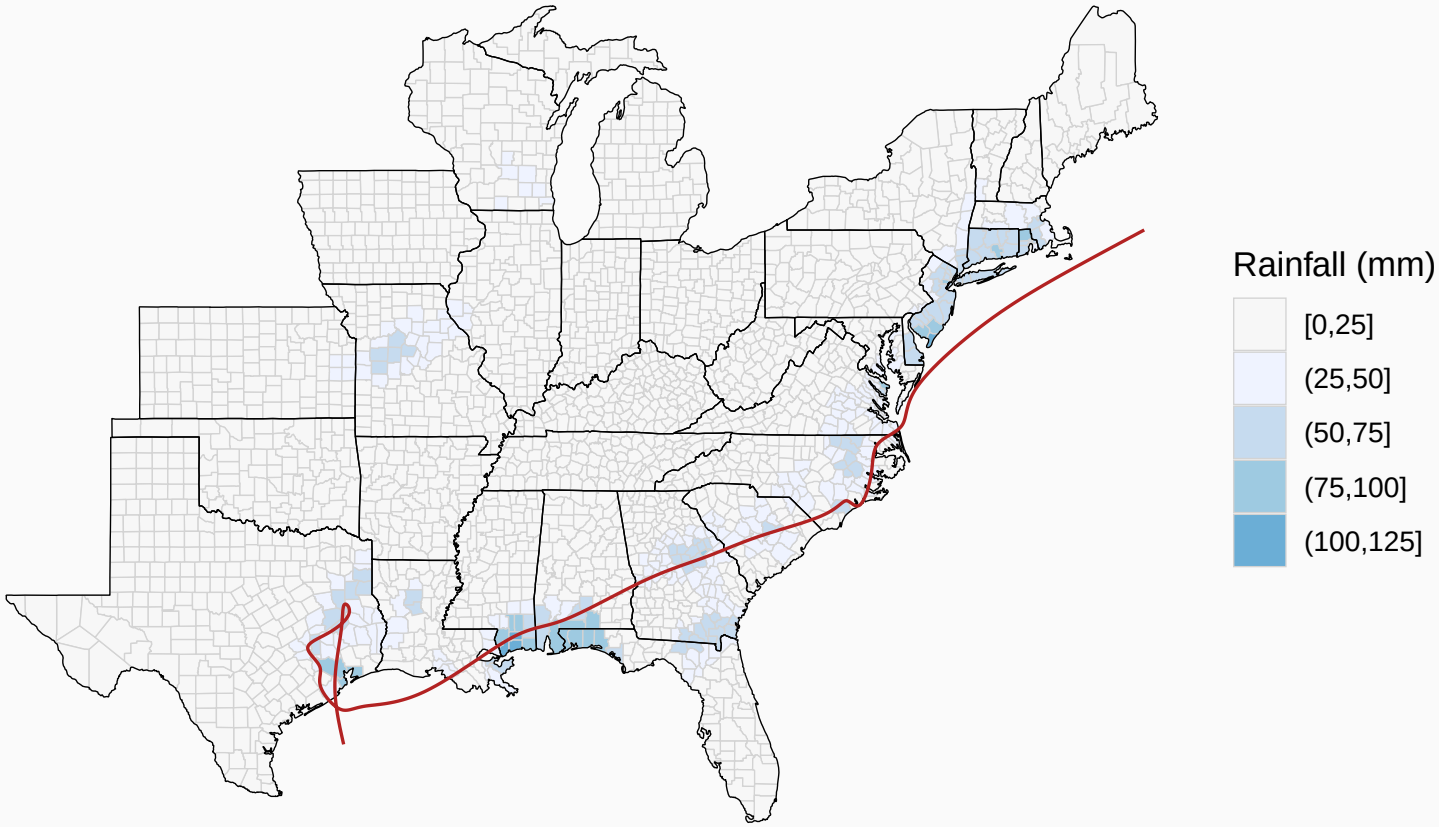
```
map_counties(storm = "Hermine-2010", metric = "rainfall")
```



Choice of days to include

Use only the storm day for the cumulative precipitation for Tropical Storm Allison.

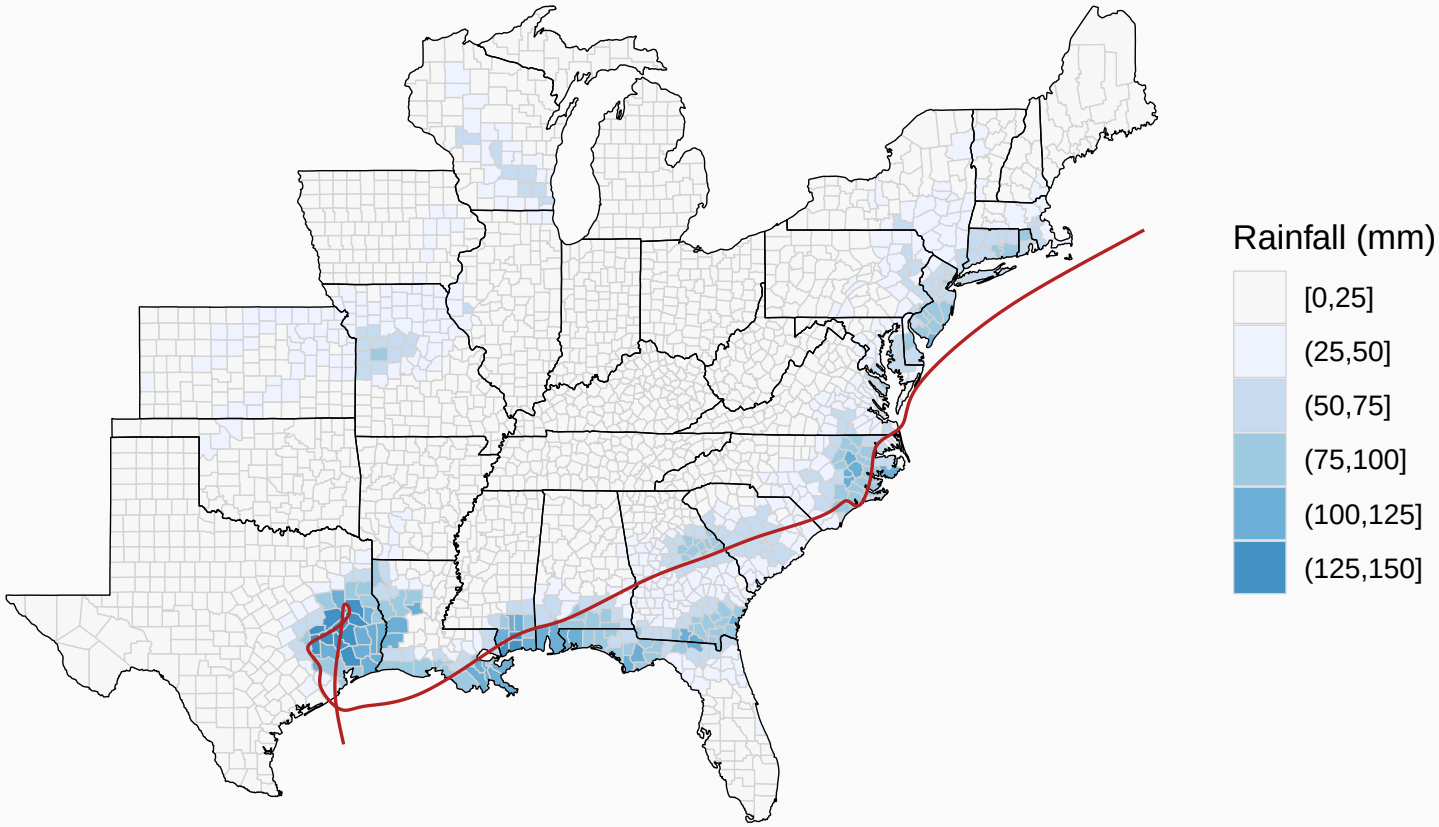
```
map_counties(storm = "Allison-2001", metric = "rainfall",
             days_included = 0)
```



Choice of days to include

Use a three-day window for the cumulative precipitation for Tropical Storm Allison.

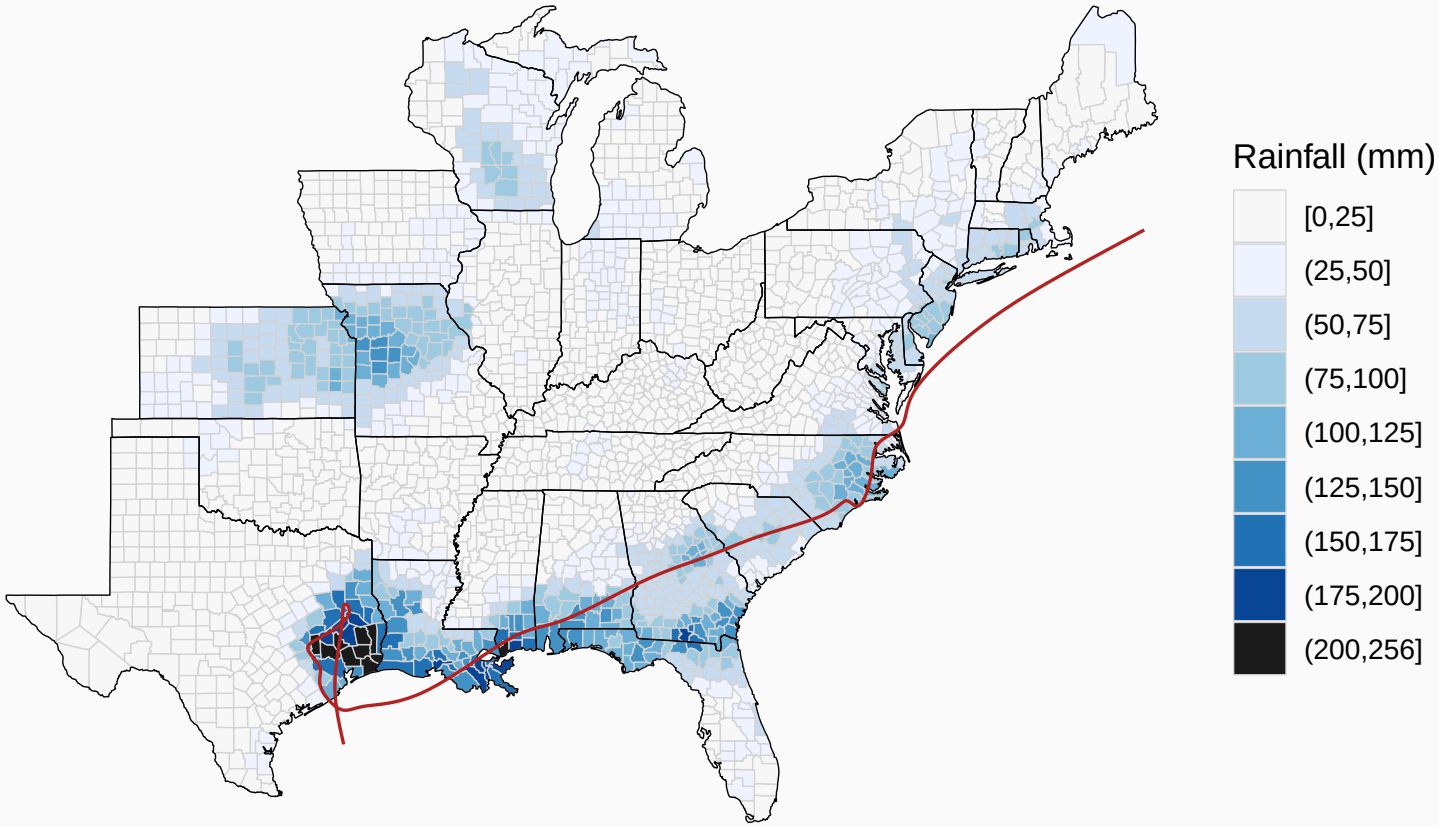
```
map_counties(storm = "Allison-2001", metric = "rainfall",
             days_included = -1:1)
```



Choice of days to include

Use a six-day window for the cumulative precipitation for Tropical Storm Allison.

```
map_counties(storm = "Allison-2001", metric = "rainfall",
             days_included = -3:3)
```

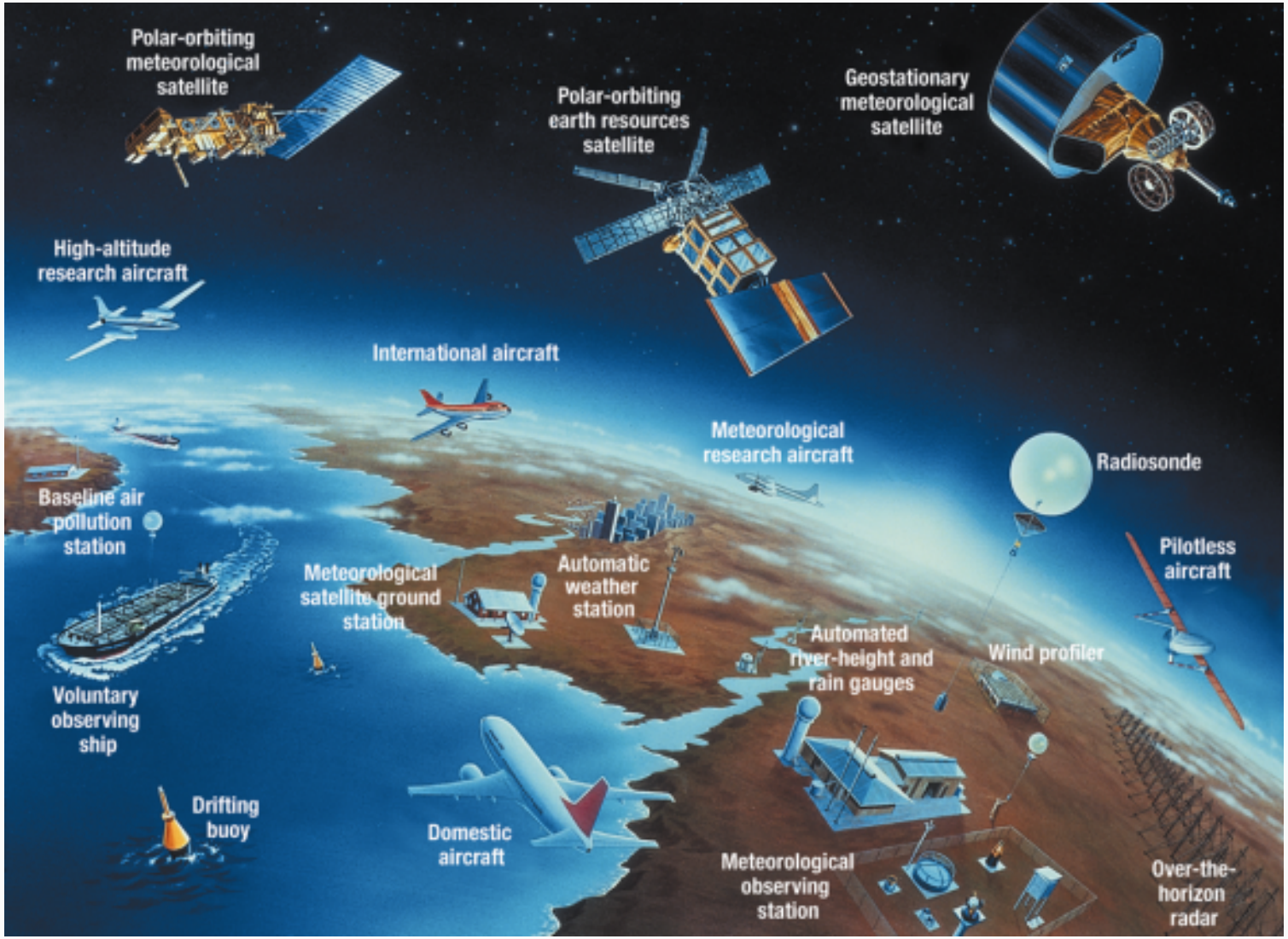


Rain data

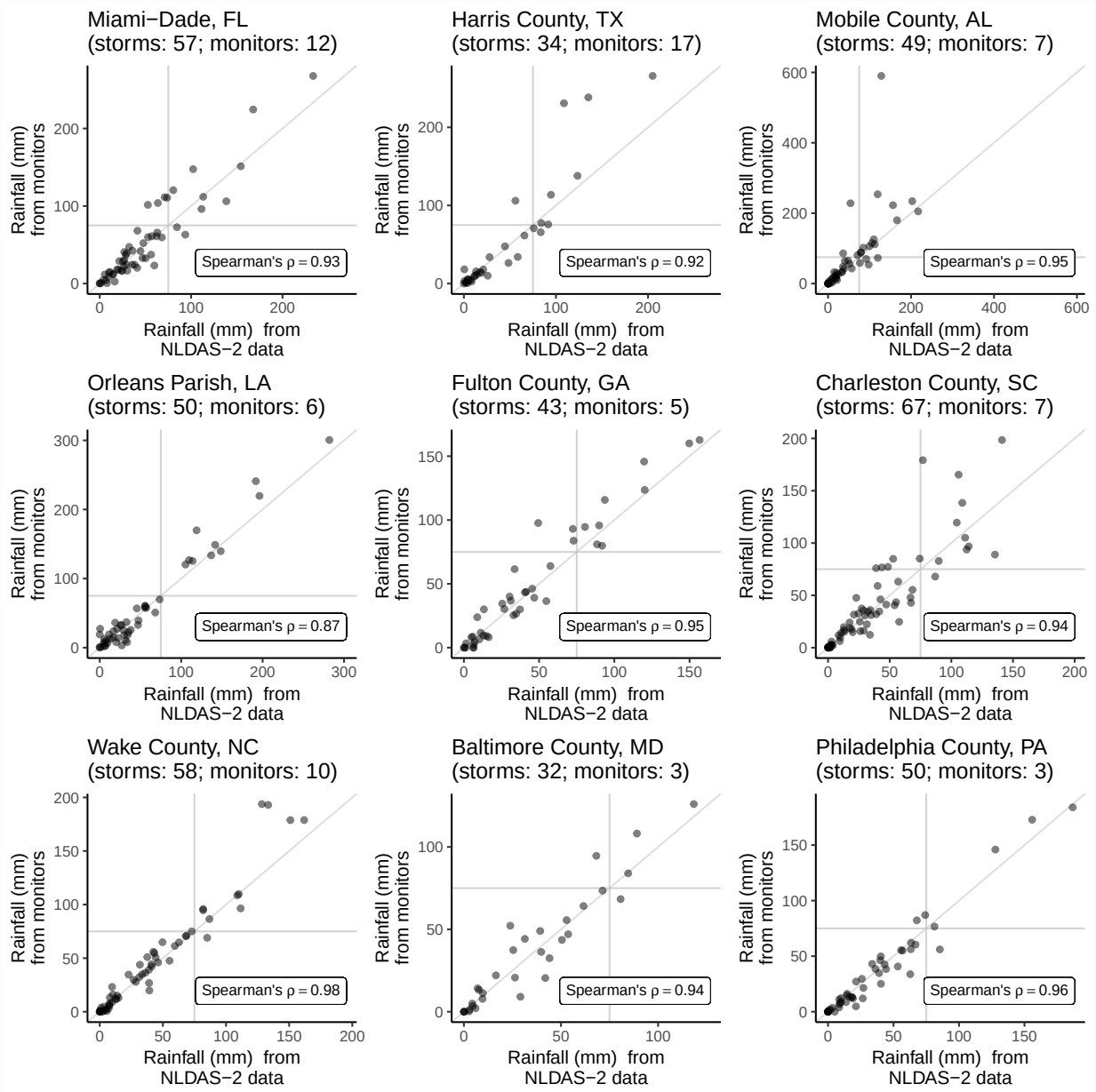
```
data("rain")
head(rain)
```

```
##    fips     storm_id lag precip precip_max
## 1 01001 Alberto-1988  -5    1.4        3.0
## 2 01001 Alberto-1988  -4    3.9        6.6
## 3 01001 Alberto-1988  -3    3.7        6.4
## 4 01001 Alberto-1988  -2    6.9        9.9
## 5 01001 Alberto-1988  -1    0.3        0.5
## 6 01001 Alberto-1988   0    1.9        3.3
```

Alternative precipitation data



Comparison with ground-based monitors

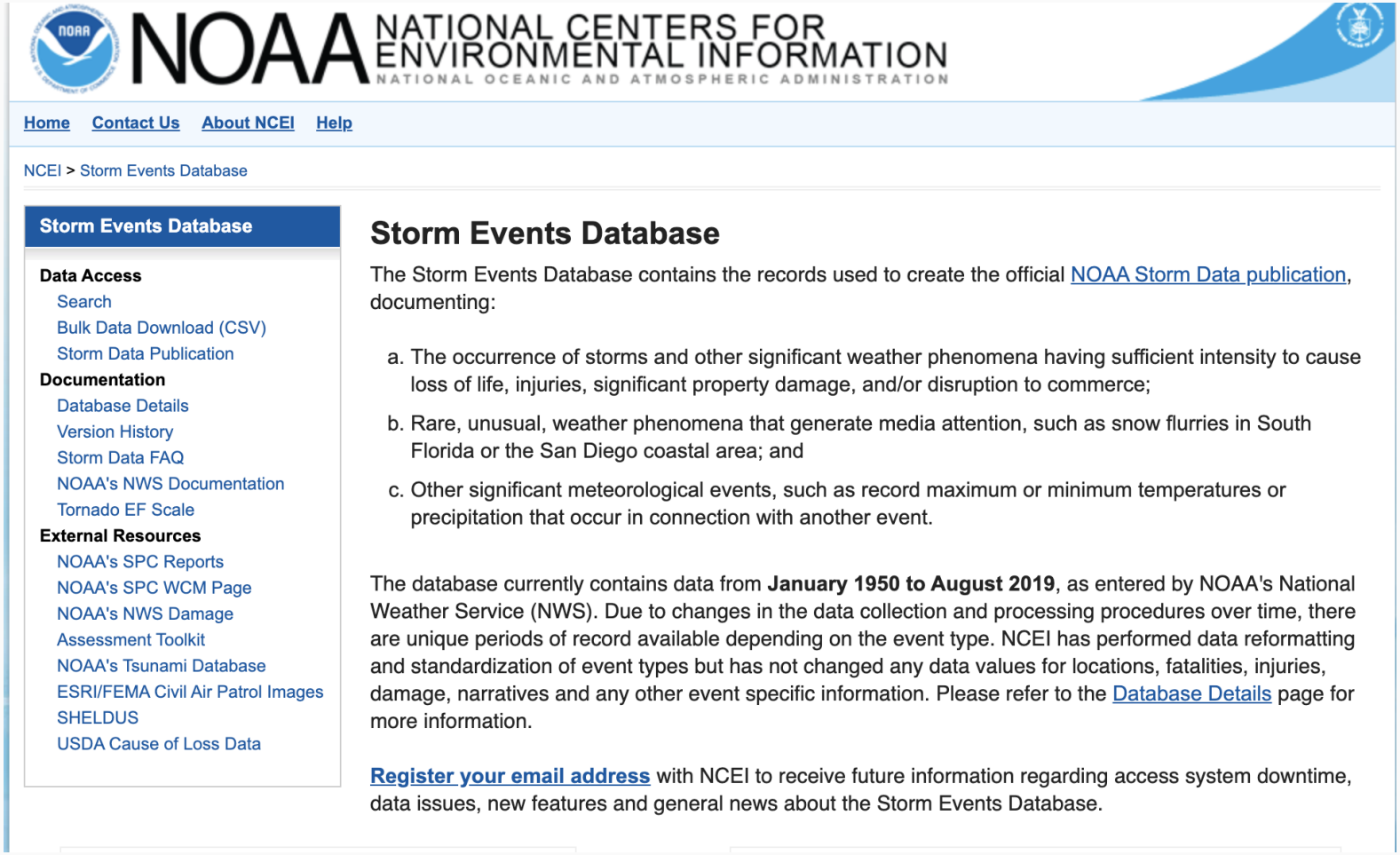


Your turn

- Map the precipitation associated with Hurricane Ike for the day of the storm.
- Map the cumulative precipitation associated with Hurricane Ike from three days before to three days after (i.e., a one week window around the storm day).

Flood and tornado exposures

NOAA Storm Events database



NOAA NATIONAL CENTERS FOR ENVIRONMENTAL INFORMATION
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

Home | Contact Us | About NCEI | Help

NCEI > Storm Events Database

Storm Events Database

Data Access
- Search
- Bulk Data Download (CSV)
- Storm Data Publication

Documentation
- Database Details
- Version History
- Storm Data FAQ
- NOAA's NWS Documentation
- Tornado EF Scale

External Resources
- NOAA's SPC Reports
- NOAA's SPC WCM Page
- NOAA's NWS Damage Assessment Toolkit
- NOAA's Tsunami Database
- ESRI/FEMA Civil Air Patrol Images
- SHELDUS
- USDA Cause of Loss Data

Storm Events Database

The Storm Events Database contains the records used to create the official NOAA Storm Data publication, documenting:

a. The occurrence of storms and other significant weather phenomena having sufficient intensity to cause loss of life, injuries, significant property damage, and/or disruption to commerce;

b. Rare, unusual, weather phenomena that generate media attention, such as snow flurries in South Florida or the San Diego coastal area; and

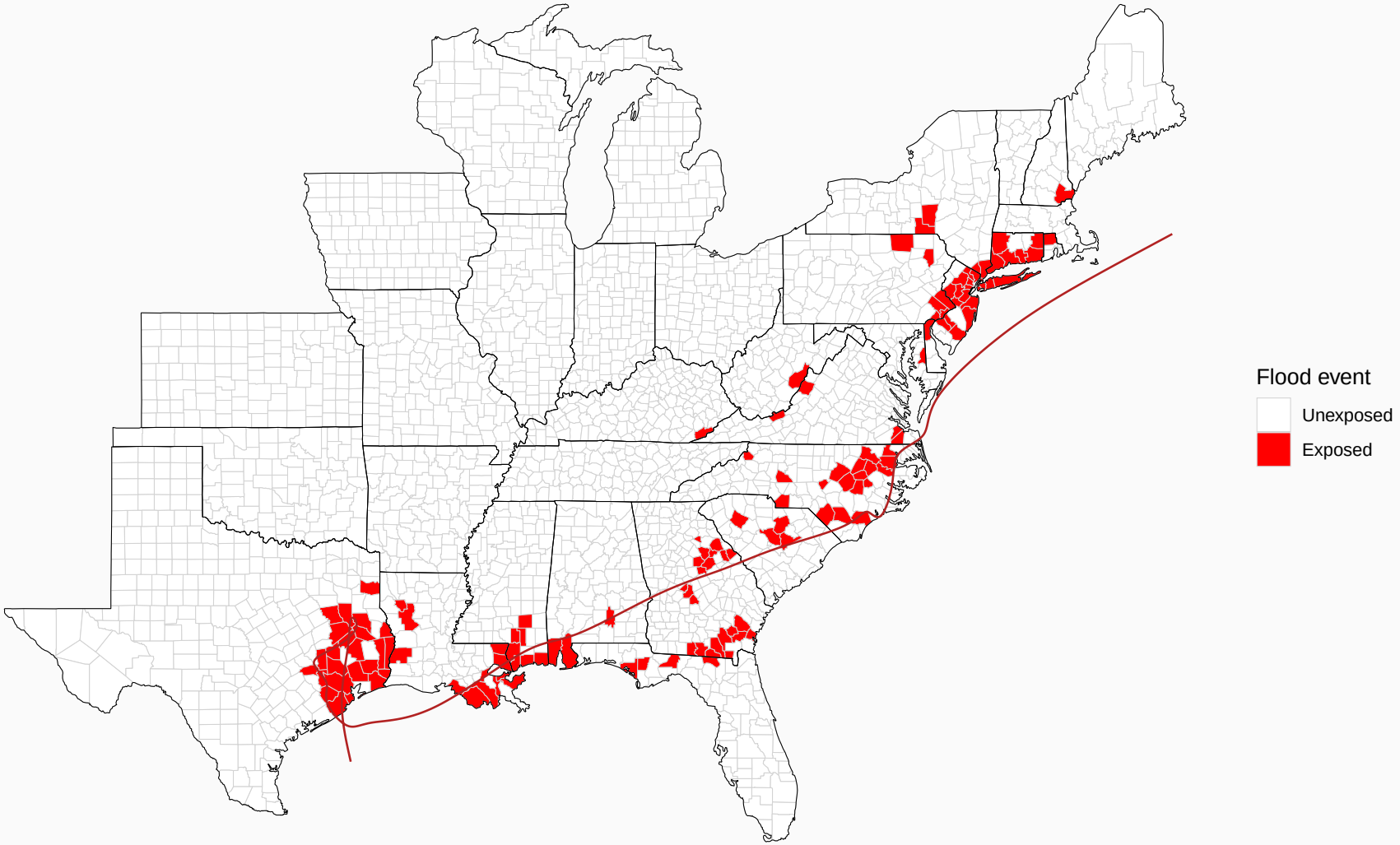
c. Other significant meteorological events, such as record maximum or minimum temperatures or precipitation that occur in connection with another event.

The database currently contains data from **January 1950 to August 2019**, as entered by NOAA's National Weather Service (NWS). Due to changes in the data collection and processing procedures over time, there are unique periods of record available depending on the event type. NCEI has performed data reformatting and standardization of event types but has not changed any data values for locations, fatalities, injuries, damage, narratives and any other event specific information. Please refer to the Database Details page for more information.

Register your email address with NCEI to receive future information regarding access system downtime, data issues, new features and general news about the Storm Events Database.

Mapping flood events for a storm

```
map_event_exposure(storm_id = "Allison-2001",
                   event_type = "flood")
```

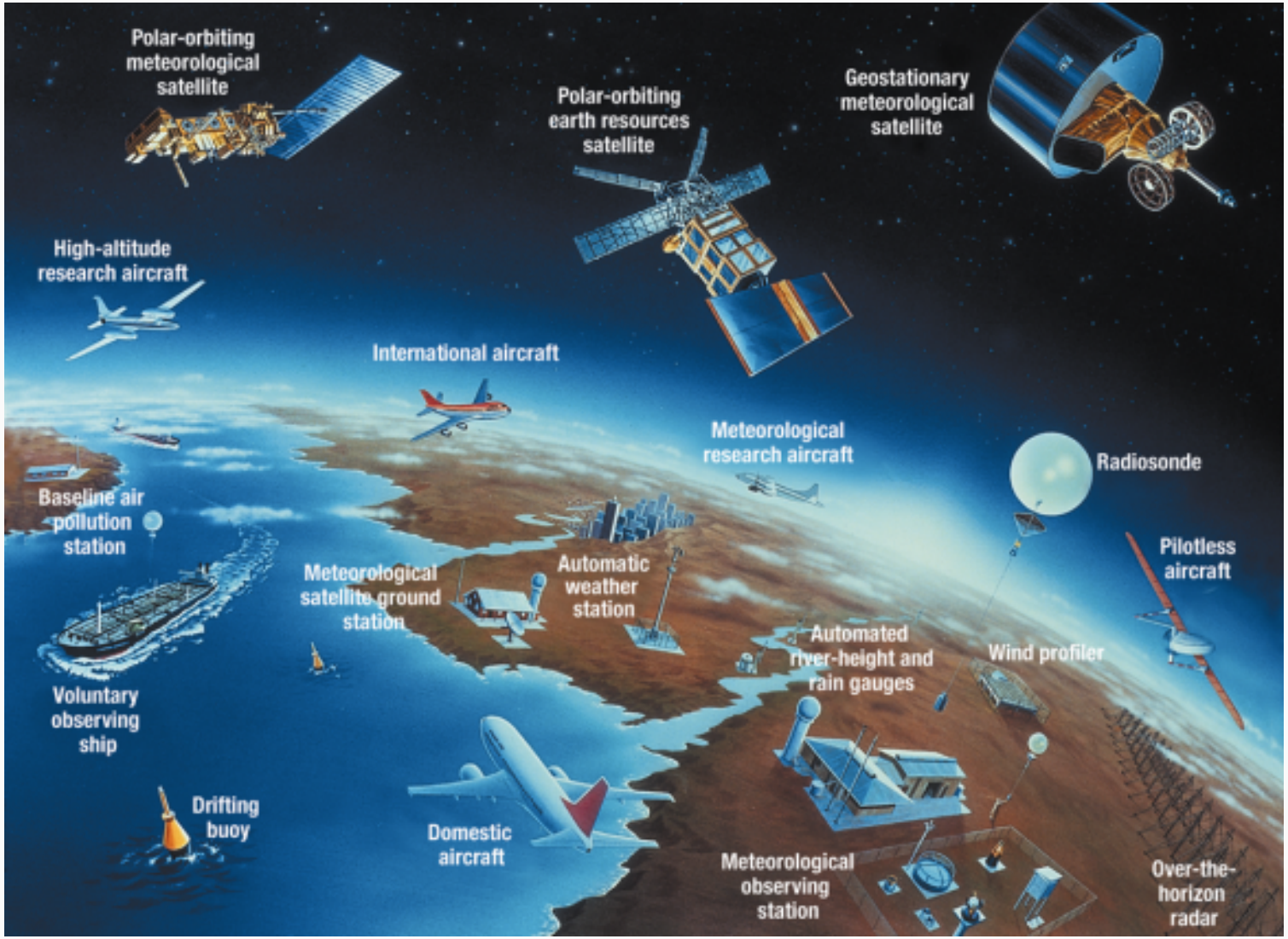


Identifying counties with flood events for a storm

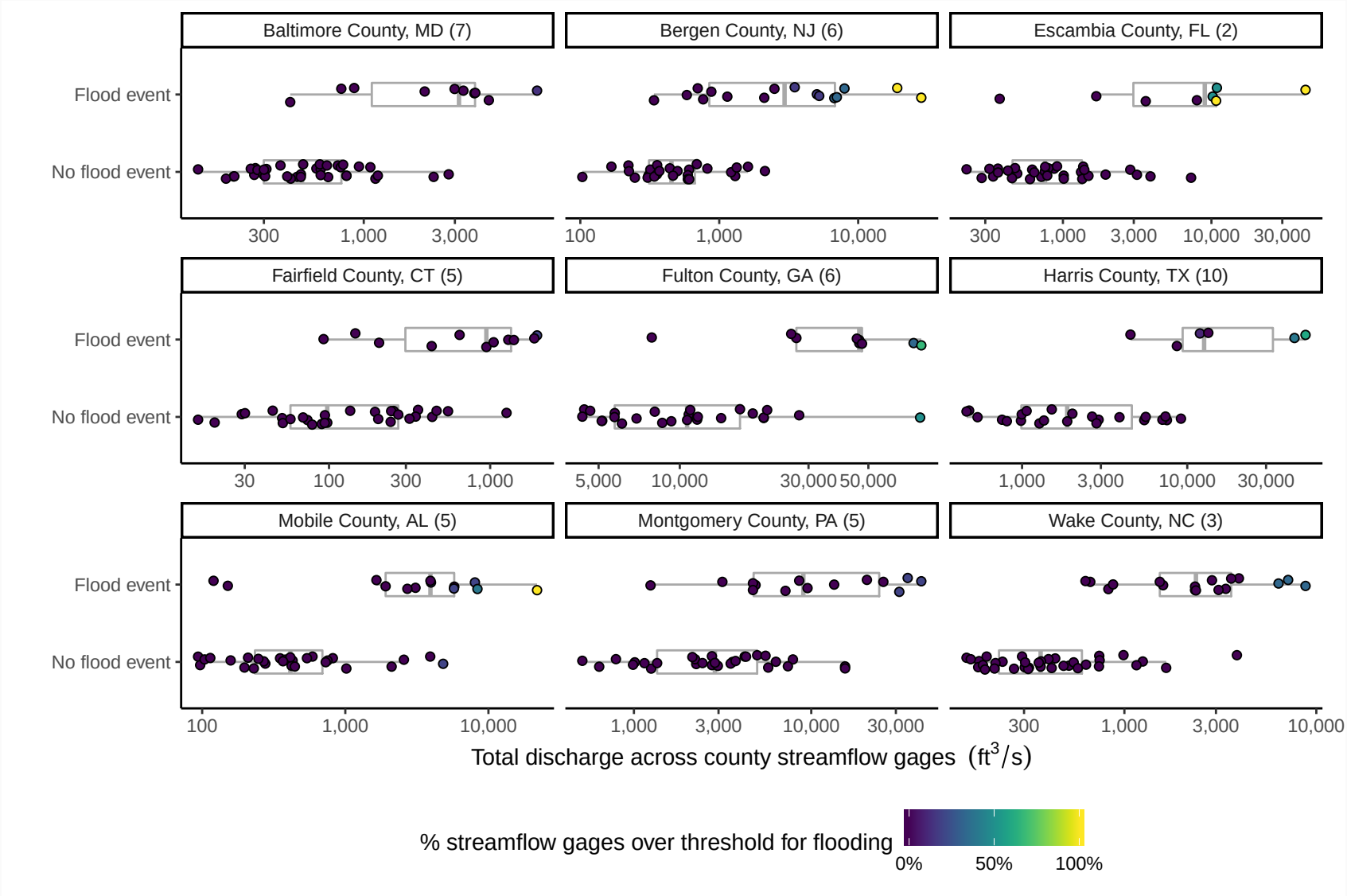
```
county_events(counties = tx_counties, event_type = "flood",
              start_year = 2001, end_year = 2001) %>%
   filter(storm_id == "Allison-2001")
```

```
##     fips     storm_id closest_time_utc storm_dist       local
## 1  48001 Allison-2001 2001-06-06 21:30  63.100572 2001-06-06
## 2  48005 Allison-2001 2001-06-07 13:00  17.287200 2001-06-07
## 3  48039 Allison-2001 2001-06-06 00:00   5.778571 2001-06-05
## 4  48073 Allison-2001 2001-06-07 00:00  28.509330 2001-06-06
## 5  48157 Allison-2001 2001-06-09 05:45  16.917932 2001-06-09
## 6  48167 Allison-2001 2001-06-06 00:45  30.537577 2001-06-05
## 7  48185 Allison-2001 2001-06-08 07:45  21.118270 2001-06-08
## 8  48199 Allison-2001 2001-06-06 07:45  91.632707 2001-06-06
## 9  48201 Allison-2001 2001-06-06 03:15  11.362299 2001-06-05
## 10 48203 Allison-2001 2001-06-07 03:15 102.434495 2001-06-06
## 11 48225 Allison-2001 2001-06-07 20:00  34.159162 2001-06-07
## 12 48241 Allison-2001 2001-06-06 11:30 109.351292 2001-06-06
## 13 48245 Allison-2001 2001-06-06 05:45 115.226972 2001-06-06
```

Other sources of flood data



Comparison with streamgage data



Your turn

- Which county had the most tropical cyclone tornado exposures between 1996 and 2015?
- Which state had the most county tropical cyclone tornado exposures between 1996 and 2015?
- Which storm brought the most tornado exposures to this state? Map county tornado exposures during this storm.

Other helpful packages

weathermetrics package

Review

Methods to Calculate the Heat Index as an Exposure Metric in Environmental Health Research

G. Brooke Anderson,[1] Michelle L. Bell,[2] and Roger D. Peng[1]

[1]Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health, Baltimore, Maryland, USA; [2]School of Forestry and Environmental Studies, Yale University, New Haven, Connecticut, USA

weathermetrics package

You can also use this package to convert among weather metrics. For example, you can convert from meters per second to knots, you can run:

```
library("weathermetrics")
speed_to_knots(c(20, 30, 40), "mps")
```

```
## [1] 38.9 58.3 77.8
```

`stormwindmodel` package

The `stormwindmodel` has the wind model used to create the county wind exposure dataset. You can use it to model winds to other geographical points, as well (e.g., ZIP code centriods).

Vignettes:

- Overview
- Full details of the storm modeling

noaastormevents package

‘noaastormevents‘

Download and explore listings from the NOAA Storm Events database. Includes the ability to pull events based on a tropical storm, using events listed close in time and distance to the storm's tracks. On CRAN.

```
library(noaastormevents)
```

```
sept_1999_events <- find_events(date_range = c("1999-09-14", "1999-09-18"))
head(sept_1999_events, 3)
```

```
## # A tibble: 3 x 14
##   begin_date end_date   state cz_type cz_name event_type source
##   <date>     <date>     <chr> <chr>   <chr>   <chr>      <chr>
## 1 1999-09-14 1999-09-14 Flor~ C       Duval   Thunderst~ TRAIN~
## 2 1999-09-14 1999-09-14 Flor~ C       St. Jo~ Thunderst~ TRAIN~
## 3 1999-09-14 1999-09-14 Ariz~ C       Marico~ Hail       OFFIC~
## # ... with 7 more variables: injuries_direct <int>,
## #   injuries_indirect <int>, deaths_direct <int>, deaths_indirect <int>,
## #   damage_property <dbl>, damage_crops <dbl>, fips <dbl>
```

noaastormevents package

You can also use this package to identify all counties with events near the time and location of a tropical cyclone:

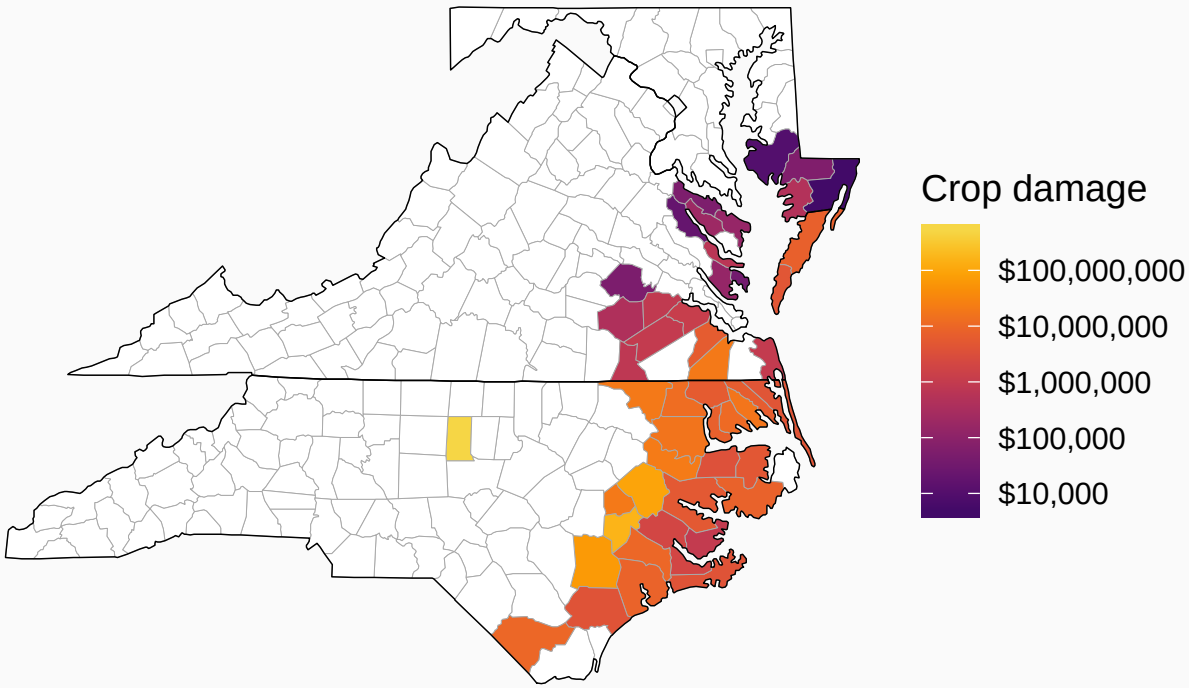
```
floyd_events <- find_events(storm = "Floyd-1999", dist_limit = 300)
head(floyd_events, 4)
```

```
## # A tibble: 4 x 15
##   begin_date end_date   state cz_type cz_name event_type source
##   <date>     <date>     <chr> <chr>   <chr>   <chr>      <chr>
## 1 1999-09-13 1999-09-14 Flor~ Z       Coasta~ Hurricane~ NEWSP~
## 2 1999-09-13 1999-09-14 Flor~ Z       Coasta~ Hurricane~ NEWSP~
## 3 1999-09-14 1999-09-14 Flor~ C       Duval   Thunderst~ TRAIN~
## 4 1999-09-14 1999-09-14 Flor~ C       St. Jo~ Thunderst~ TRAIN~
## # ... with 8 more variables: injuries_direct <int>,
## #   injuries_indirect <int>, deaths_direct <int>, deaths_indirect <int>,
## #   damage_property <dbl>, damage_crops <dbl>, fips <dbl>, storm_id <chr>
```

noaastormevents package

You can also pull out and map specific information in this database, including specific types of disasters, property damage, and crop damage.

```
floyd_events %>%
  map_events(plot_type = "crop damage",
             states = c("north carolina", "virginia", "maryland"))
```



futureheatwaves package

Working with Daily Climate Model Output Data in R and the futureheatwaves Package

by G. Brooke Anderson, Colin Eason, and Elizabeth A. Barnes

Abstract Research on climate change impacts can require extensive processing of climate model output, especially when using ensemble techniques to incorporate output from multiple climate models and multiple simulations of each model. This processing can be particularly extensive when identifying and characterizing multi-day extreme events like heat waves and frost day spells, as these must be processed from model output with daily time steps. Further, climate model output is in a format and follows standards that may be unfamiliar to most R users. Here, we provide an overview of working with daily climate model output data in R. We then present the **futureheatwaves** package, which we developed to ease the process of identifying, characterizing, and exploring multi-day extreme events in climate model output. This package can input a directory of climate model output files, identify all extreme events using customizable event definitions, and summarize the output using user-specified functions.

futureheatwaves package

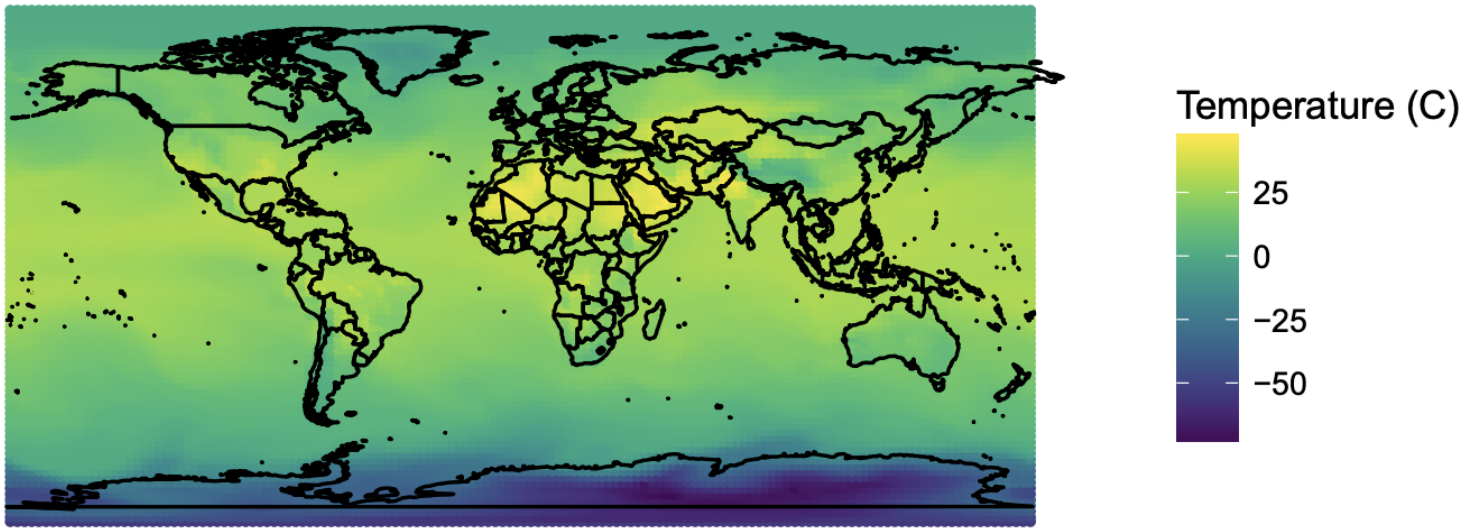


Figure 2: Example of mapping near-surface air temperature data worldwide for a single day of climate model output data. This map uses data from the Geophysical Fluid Dynamics Laboratory's Earth System Model 2G, r1i1p1 ensemble member, on a single day in the summer of 2075. Full code for recreating the map is available in the "starting_from_netcdf" vignette of the **futureheatwaves** package.