Lights Out in Houston: Mapping the Effects of Winter Storm Uri

Vedika Shirtekar

11/10/25

Primary Objective

Winter Storm Uri was a powerful and catastrophic winter storm that devastated several communities in Texas during February 13-17 2021. The state was unprepared for a significant loss of power; power plants and grids were not prepared to handle freezing temperatures coupled with increased electricity usage to heat homes (Zhou et al., 2024). Nearly 246 people perished, 4.5 million homes lost power, and costed the state about $195 million in damages (Zhou et al., 2024). As such, Winter Storm Uri is listed as the costliest and one of the deadliest natural disasters in Texan history.

The City of Houston, the county's most populous city, had an estimated population of 2,333,346 in 2023 with a median household income of $62,894 (DataUSA, n.d.) . The city experienced a population growth of approximately 0.18% and a median household income increase of about 4.06% (DataUSA, n.d.). Houston was one of several urban centers in Harris County that were significantly affected by the storm and is a key area for analysis.

The purpose of this assignment is to assess the extent to which houses and census tracts in Houston lost power during Winter Storm Uri. The analysis utilizes spatial manipulation between raster and vector types to identify homes in Houston that experienced a blackout, as well as an assessment of median income for census tracts affected by the blackout based on socioeconomic data. The following research question was referenced to guide the analysis:

To what extent were homes and census tracts in Houston impacted by the power outage caused by Winter Storm Uri?

Load in required packages

1. Load in the appropriate packages for the analysis.

```
# Import packages
library(here) # Load "here" to locate and reference files
library(tidyverse) # Load the tidyverse" for data cleaning
library(sf) # Load "sf" for GIS analysis
library(raster) # Load "raster" for accessing raster data types
library(ggplot2) # Load "ggplot2" for data visualization
library(tmap) # Load "tmap" for functions to create and layer maps
library(kableExtra) # Load "kableExtra" for table formatting
library(stars) # Load "stars" for integration with "sf"
```

Part 1: Create a Blackout Mask

To determine the areas affected by blackouts, a mask was generated to classify each cell based on whether or not it experienced a blackout.

2. Read in each `lights` raster using `read_stars()`. Both pre- and post- storm days have two tiles per date that must be loaded in and combined.

```
# Read in each raster file as a stars object

# Feb 7
lights1 <- read_stars(here::here("data",
                                 "VNP46A1",
                                 "VNP46A1.A2021038.h08v05.001.2021039064328.tif"))

# Feb 7
lights2 <- read_stars(here::here("data",
                                 "VNP46A1",
                                 "VNP46A1.A2021038.h08v06.001.2021039064329.tif"))

# Feb 16
lights3 <- read_stars(here::here("data",
                                 "VNP46A1",
                                 "VNP46A1.A2021047.h08v05.001.2021048091106.tif"))

# Feb 16
lights4 <- read_stars(here::here("data",
                                 "VNP46A1",
                                 "VNP46A1.A2021047.h08v06.001.2021048091105.tif"))
```

3. Use conditional statements to verify that each `lights` tile has the same CRS as `lights1`. If mismatches are found, issue explicit warnings and transform the tile CRS to the reference CRS (`lights1`).

```
# Do the CRS match? Use series of if statements with warnings
# Create list of spatial objects
spatial_objects <- list(
  lights1 = lights1,
  lights2 = lights2,
  lights3 = lights3,
  lights4 = lights4
)

# Use lights1's CRS as reference
ref_crs <- st_crs(spatial_objects$lights1)
```

```
# Check and transform each tile with if/else statements
if (st_crs(spatial_objects$lights2) != ref_crs) {
  warning("lights2 CRS does not match.
          Transforming to match lights1 CRS.")
  spatial_objects$lights2 <- st_transform(spatial_objects$lights2, ref_crs)
} else {
  message("lights2 CRS already matches lights1 CRS.")
}
```

```
lights2 CRS already matches lights1 CRS.
```

```
# Check and transform each tile with if/else statements
if (st_crs(spatial_objects$lights3) != ref_crs) {
  warning("lights3 CRS does not match.
          Transforming to match lights1 CRS")
  spatial_objects$lights3 <- st_transform(spatial_objects$lights3, ref_crs)
} else {
  message("lights3 CRS already matches lights1 CRS.")
}
```

```
lights3 CRS already matches lights1 CRS.
```

```r
# Check and transform each tile with if/else statements
if (st_crs(spatial_objects$lights4) != ref_crs) {
  warning("lights4 CRS does not match. Transforming
          to match lights1 CRS")
  spatial_objects$lights4 <- st_transform(spatial_objects$lights4, ref_crs)
} else {
  message("lights4 CRS already matches lights1 CRS.")
}
```

```
lights4 CRS already matches lights1 CRS.
```

4. Combine each tile raster into a single night light raster using `st_mosaic()` for each day.

```r
# Combine tiles of same day using st_mosaic()
lights_day1 <- st_mosaic(lights1, lights2)  # Feb 7
lights_day2 <- st_mosaic(lights3, lights4) # Feb 16
```

5. Calculate the change in night light intensity, presumed to result from the storm, and store the results in a raster. Reclassify the difference raster so that areas with a drop greater than 200 nW cm ^{2} sr ^{1} are identified as experiencing a blackout.

```r
# Compute difference (Feb 7 - Feb 16)
diff_raster <- lights_day1 - lights_day2

# Double check class of diff_raster
class(diff_raster)
```

```
[1] "stars"
```

```r
#Compare values from February 7 to February 16
# Store diff_raster in blackout_mask to create copy
blackout_mask<- diff_raster

# Assign NAs for differences less than positive 200 (inverted drop)
blackout_mask[blackout_mask < 200] <- NA
```

6. Vectorize the blackout mask using `st_as_sf()` and ensure geometries are valid prior to cropping to the bounding box extent.

```r
# Vectorize blackmask
blackout_mask <- st_as_sf(blackout_mask)
blackout_mask <- st_make_valid(blackout_mask)

# Ensure blackout_mask is an sf object
# Use all() to check for whether class() contains "sf" AND "data.frame"
if (all(c("sf", "data.frame") %in% class(blackout_mask))){
  # Pass a message to ensure black_outmask is an sf object
  message("blackout_mask is an sf object and is ready for cropping.")
} else{ # Stop running if not an sf object (still stars object)
  stop("blackout_mask is not an sf object. Double check st_as_sf() output.")
}
```

```
blackout_mask is an sf object and is ready for cropping.
```

7. Create a bounding box using `st_bbox()` for the extent of Houston for coordinates: (-96.5, 29), (-96.5, 30.5), (-94.5, 30.5), and (-94.5, 29).

```r
# Create bounding box using st_bbox()
houston_bbox <- st_bbox(c(xmin = -96.5, # Min longitude
                          xmax = -94.5, # Max longitude
                          ymin = 29,    # Min latitude
                          ymax = 30.5), # Max latitude
                        crs = 4326) # Use EPSG code for CRS

# Double chcek whether CRS is a match
if (st_crs(houston_bbox) != st_crs(blackout_mask)) {
  warning("houston_bbox CRS does not match.
          Transforming to match blackout_mask CRS")
  # Transform to match blackout_mask CRS
  houston_bbox <- st_transform(houston_bbox, st_crs(blackout_mask))
} else {
  message("houston_bbox CRS already matches blackout_mask CRS.")
}
```

```
houston_bbox CRS already matches blackout_mask CRS.
```

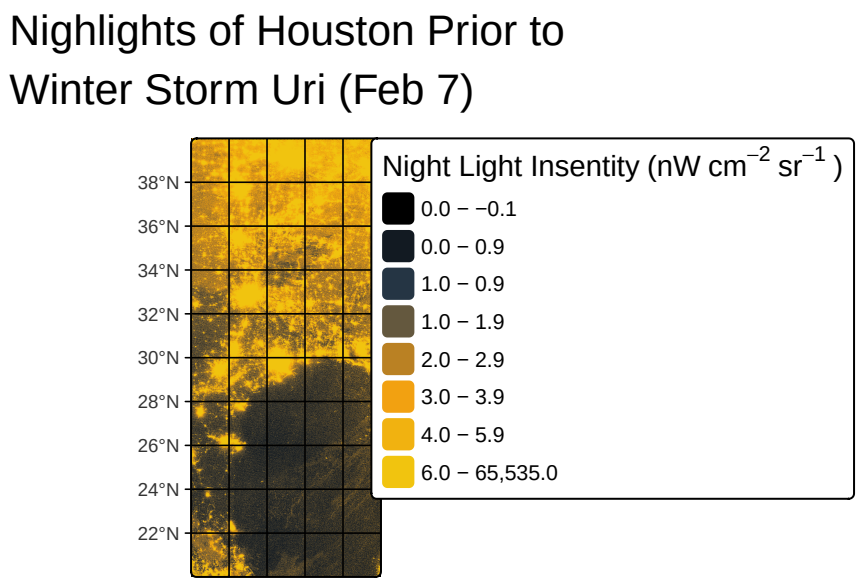
8. Because `houston_bbox` is a `bbox`, it is necessary to convert it to an `sf` spatial type prior to cropping. Convert `houston_bbox` to an `sf` polygon type, then use `st_crop()` to crop (spatially subset) the blackout mask to the Houston area.

```r
# Convert boundaries of Houston to sf polygon
houston <- st_as_sfc(houston_bbox)
# Render valid geometry for Houston polygon
houston <- st_make_valid(houston)

# Crop blackout mask to Houston boundaries
houston_blackout <- st_crop(blackout_mask, houston)

# Reproject cropped blackout dataset to EPSG:3083
# (NAD83 / Texas Centric Albers Equal Area)
houston_blackout <- st_transform(houston_blackout, crs = "EPSG:3083")
```

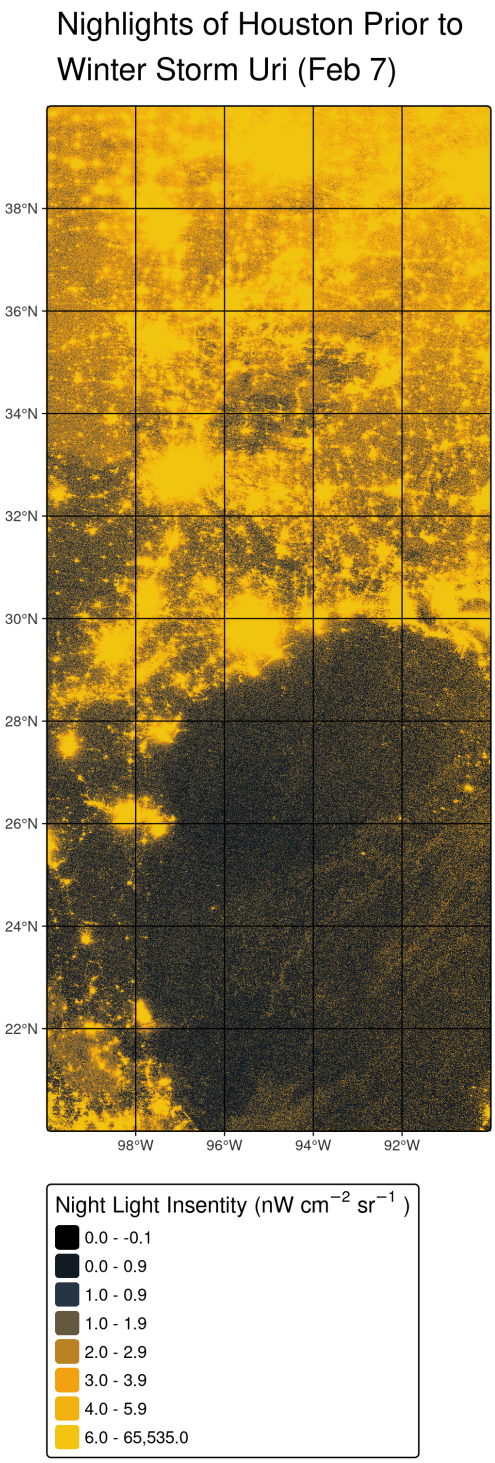
9. Create two maps using the combine raster tiles of the pre- and post- storm days to compare night light intensities.

```r
# Pre-storm lights
prestorm <- tm_shape(lights_day1) + # Feb 7 tiles
  tm_raster(
    col.scale  = tm_scale_intervals(
      style = "quantile", # Bin intensity range
      n = 8, # For binning with four colors
      values = c("black", # Black to orange custom color palette
                 "#2c3e50",
                 "#f39c12",
                 "#f1c40f")),
      col.legend = tm_legend( # Use expression() for superscript formatting
        expression(
          "Night Light Insentity"~
          "(nW"~cm^-2~ sr^-1~")"))) +

  tm_graticules( # Add graticules to establish latitude and longitude network
    col = "black",
    lwd =.8) + # Establish "thickness" of lines

  tm_layout( # Center title outside bounding box
    main.title = "Nighlights of Houston Prior to \nWinter Storm Uri (Feb 7)",
    legend.outside = TRUE, # Place legend outside map frame
    legend.outside.position = "right", # Place legend to right
    component.autoscale = FALSE, # Disable autoscaling for title
  )
```

```
# View in cell block output
prestorm
```



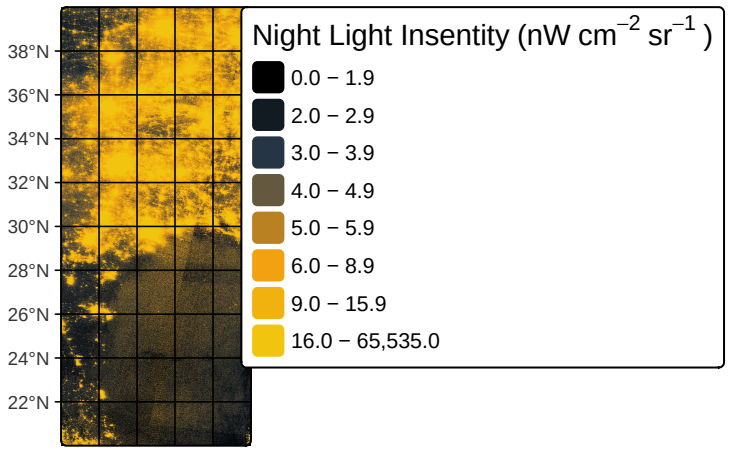

```
# Save finalized map to figs
tmap_save(prestorm, "figs/pre_storm.png")
```


Nighlights of Houston Prior to
Winter Storm Uri (Feb 7)
38°N
36°N
34°N
32°N
30°N
28°N
26°N
24°N
22°N
98°W
96°W
94°W
92°W
Night Light Insentity (nW cm^{-2} sr^{-1})
0.0 - -0.1
0.0 - 0.9
1.0 - 0.9
1.0 - 1.9
2.0 - 2.9
3.0 - 3.9
4.0 - 5.9
6.0 - 65,535.0

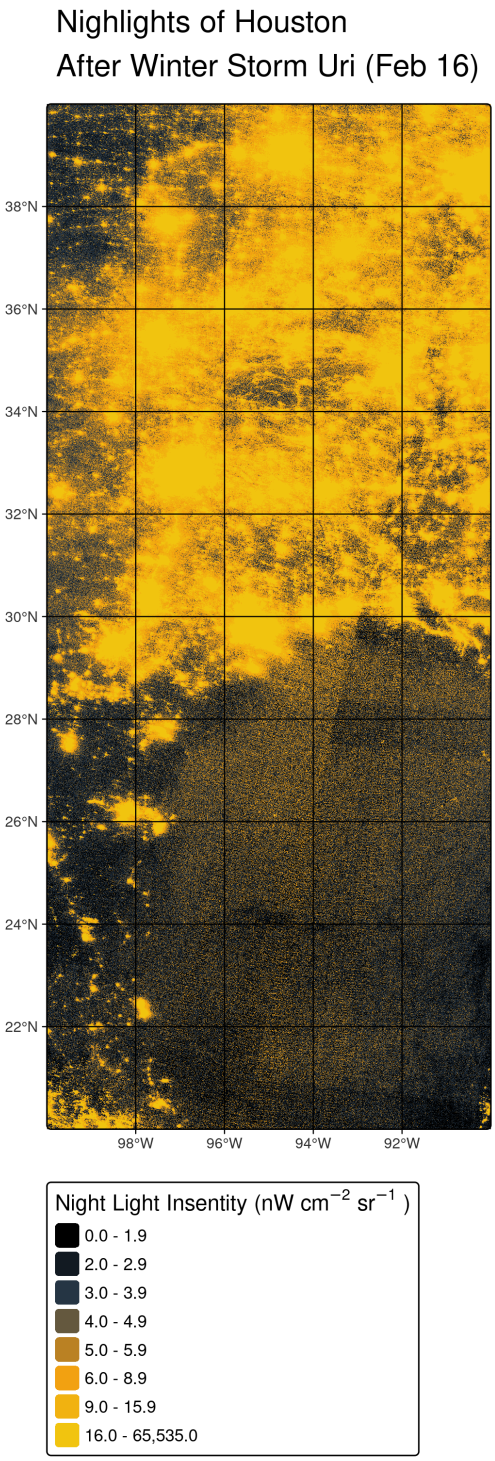
Map 1. Finalized Version of Nightlights of Houston prior to Winter Storm Uri (Feb 7).

```
# Post-storm lights
poststorm <- tm_shape(lights_day2) + # Feb 16 tiles
  tm_raster(
    col.scale  = tm_scale_intervals(
      style = "quantile", # Bin intensity range
      n = 8, # For binning with four colors
      values = c("black", # Black to orange custom color palette
                 "#2c3e50",
                 "#f39c12",
                 "#f1c40f")),
      col.legend = tm_legend( # Use expression() for superscript formatting
        expression(
          "Night Light Insentity"~
          "(nW"~cm^-2~ sr^-1~")"))) +

  tm_graticules( # Add graticules to establish latitude and longitude network
    col = "black",
    lwd =.8) + # Establish "thickness" of lines

  tm_layout( # Center title outside bounding box
    main.title = "Nighlights of Houston \nAfter Winter Storm Uri (Feb 16)",
    legend.outside = TRUE, # Place legend outside map frame
    legend.outside.position = "right", # Place legend to right
    component.autoscale = FALSE, # Disable autoscaling for title
  )

poststorm
```

Nighlights of Houston
After Winter Storm Uri (Feb 16)




```
# Save finalized map to figs
tmap_save(poststorm, "figs/post_storm.png")
```


Nighlights of Houston
After Winter Storm Uri (Feb 16)
38°N
36°N
34°N
32°N
30°N
28°N
26°N
24°N
22°N
98°W
96°W
94°W
92°W
Night Light Insentity (nW cm^{-2} sr^{-1})
0.0 - 1.9
2.0 - 2.9
3.0 - 3.9
4.0 - 4.9
5.0 - 5.9
6.0 - 8.9
9.0 - 15.9
16.0 - 65,535.0

Map 2. Finalized Version of Nightlights of Houston After Winter Storm Uri (Feb 16).

Part 2: Excluding highways from the blackout mask

It is possible highways may have experienced changes in night light intensities that were not related to the storm. This part of the analysis focuses on excluding locations within 200 meters of all highways in Houston.

10. Load in the highways (`roads`) geopackage and use a query to filter for motorways (highways) among other road types.

```
# Read in only highways data using a search query
roads <- read_sf(here::here("data", "gis_osm_roads_free_1.gpkg"),
                 query = "SELECT * FROM
                 gis_osm_roads_free_1 WHERE fclass='motorway'")
```

11. Use `st_union()` on `roads` to dissolve boundaries between adjacent highways in Houston.

```
# Update roads to include dissolved highways
roads <- st_union(roads)
```

12. Ensure the CRS of `roads` and `houston_blackout` match.

```
# Double check CRS match between roads and houston_blackout
if (st_crs(roads) != st_crs(houston_blackout)) {
  warning("roads CRS does not match. Transforming
          to match houston_blackout CRS")
  roads <- st_transform(roads, crs = st_crs(houston_blackout))
} else {
  message("roads CRS already matches houston_blackout CRS.")
}
```

```
Warning: roads CRS does not match. Transforming
          to match houston_blackout CRS
```

13. Create a 200 meter buffer around all highways using `st_buffer()`. Then, apply `st_union()` to dissolve the individual highway buffers into a single continuous polygon.

```r
# Ensure units for roads are in meters
print(paste("Units of `roads`:", st_crs(roads)$units))
```

```
[1] "Units of `roads`: m"
```

```r
# Create a 200 meter buffer around highways
roads_buffer_200 <- st_buffer(roads, dist = 200)

# Dissolve individual buffers to form continuous polygon
# for areas within 200 meters of highways
highways_200 <- st_union(roads_buffer_200)
```

14. Use `st_filter()` to compare the blackout polygons (`houston_blackout`) with the 200 meter highway buffer (`highways_200`). Set `.predicate = st_disjoint` to select only blackout areas that are (spatially) separate from the buffered highways, as well as removing any overlapping regions.

```r
# Identify areas of blackout that do NOT overlap highways
blackout_not_highway <- st_filter(houston_blackout, highways_200,
                                  .predicate = st_disjoint)
```

Part 3: Identify the number of homes likely impacted by blackouts

In the section, homes that overlap with areas that experienced blackouts were identified by combining the locations of homes and areas that experienced a blackout (`houston_blackout`).

15. Read in the `buildings` geopackage and pass a query to only select for residential areas, apartments, houses, carvans, and 'detached' living.

```r
# Load in buildings (homes)
buildings <- read_sf(
  here::here("data", "gis_osm_buildings_a_free_1.gpkg"),
  query = "
    SELECT *
    FROM gis_osm_buildings_a_free_1
    WHERE (type IS NULL AND name IS NULL) OR type IN ('residential', 'apartments',
    'house', 'static_caravan', 'detached')")
```

16. Selects all building features that intersect areas of blackout not overlapping highways, using `st_filter()` with the `st_intersects` predicate. Then, create a map showcasing the affected homes in Houston.

```
# Ensure CRS of buildings and blackout_not_highway match
if (st_crs(buildings) != st_crs(blackout_not_highway)) {
  warning("buildings CRS does not match. Transforming
          to match blackout_not_highway CRS")
  buildings <- st_transform(buildings, crs = st_crs(blackout_not_highway))
} else {
  message("buildings CRS already matches blackout_not_highway CRS.")
}
```

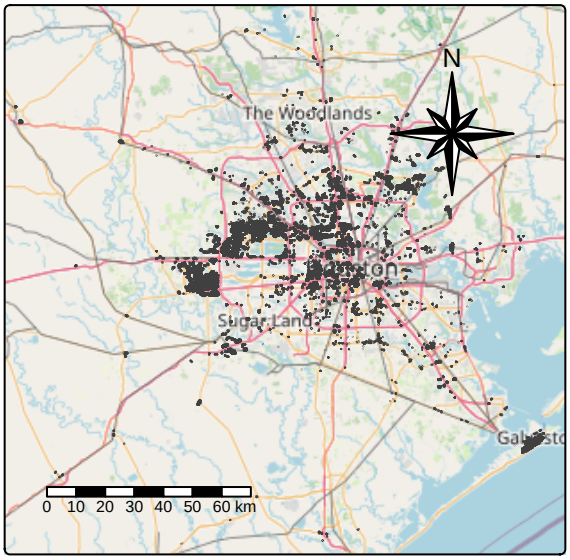
```
Warning: buildings CRS does not match. Transforming
          to match blackout_not_highway CRS
```

```
# Obtain intersecting (attribute) features of buildings
# and blackout_not_highway
buildings_affected_blackout <- st_filter(buildings, blackout_not_highway,
                                         .predicate = st_intersects)

# Identify the number of impacted homes
print(paste("About",
            # Count the number of rows and round to nearest whole number
            round(nrow(buildings_affected_blackout), 0),
            "homes were impacted by the blackout."))
```

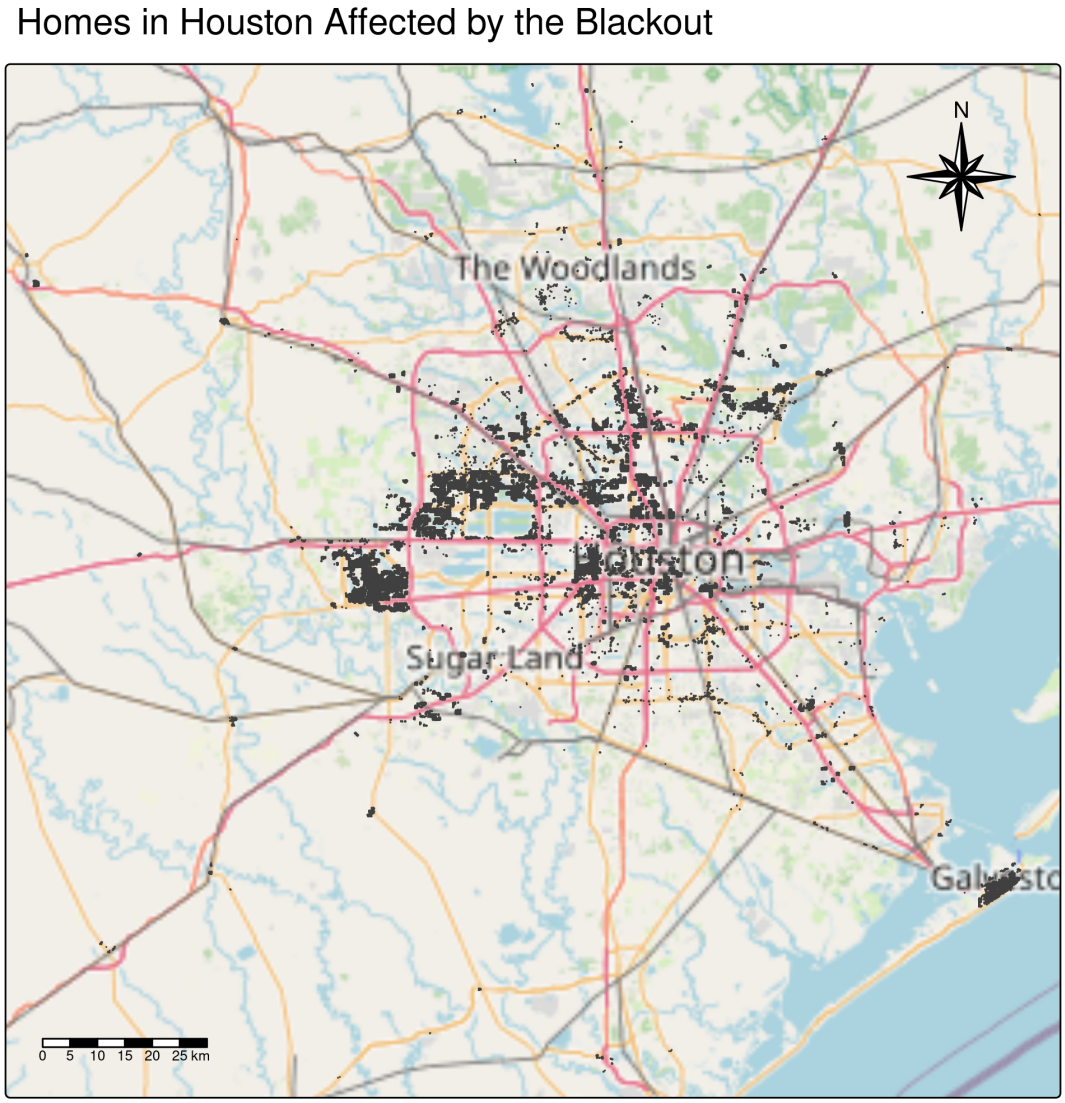
```
[1] "About 139148 homes were impacted by the blackout."
```

```
# Create map for homes affected vs non affected by the blackout
affected_homes_map <- tm_shape(buildings_affected_blackout) +

  tm_polygons() + # Set polygons for vector type

  tm_basemap(server = "OpenStreetMap") + # Establish basemap

   tm_scale_bar( # Add scale bar for scale
    position = c("left", "bottom")) +
```

```
   tm_compass( # Add compass for orientation
    type = "8star",
    position = c("right", "top"),
  ) +

  tm_layout(
    main.title = "Homes in Houston Affected by the Blackout",
    main.title.position = "center") # Center title

# View map in cell block output
affected_homes_map
```




```
# Save finalized map to figs
tmap_save(affected_homes_map,
          "figs/Homes_in_Houston_Affected_by_Blackout.png")
```



Map 3. Finalized Version of homes in Houston impacted by the winter storm blackout.

Part 4: Identify impacted census tracts

Here, socioeconomic data was obtained from the `ACS_2019_5YR_TRACT_48_TEXAS` geodatabase by joining geometric data for Texas with attribute data describing median income. Census tracts impacted by the blackout were mapped, and a comparison of the median income of census tracts impacted were visualized for further interpretation.

17. Read the geometry layer (`ACS_2019_5YR_TRACT_48_TEXAS`) and the attribute layer (`X19_INCOME`) from the geodatabase using `st_layer()` to identify available layers. Then, perform a left join on the shared `GEOID` column (with different names in each layer) to merge the spatial geometries with corresponding income data.

```
# Utilize st_layers() to examine the INCOME and TEXAS layers
#st_layers(here("data", "ACS_2019_5YR_TRACT_48_TEXAS.gdb"))

# Read in geometric layer (Texas)
geometry_socio <- st_read(here("data", "ACS_2019_5YR_TRACT_48_TEXAS.gdb"),
                      layer = "ACS_2019_5YR_TRACT_48_TEXAS")

# Read in attribute data for income
attributes_socio <- st_read(here("data", "ACS_2019_5YR_TRACT_48_TEXAS.gdb"),
                        layer = "X19_INCOME")

# Perform a left join for combining spatial and income data
socioeconomic <- geometry_socio %>%
  left_join(attributes_socio,
            by = c("GEOID_Data" = "GEOID")) # Match column names to join on GEOID_Data
```

18. Link building locations to ACS census tracts by using `st_filter()` with the `st_intersects` predicate to identify tracts containing affected homes.

```
# Ensure CRS of socioeconomic and buildings_affected_blackout match
if (st_crs(socioeconomic) != st_crs(buildings_affected_blackout)) {
  warning("socioeconomic CRS does not match. Transforming
          to match buildings_affected_blackout CRS")
  socioeconomic <- st_transform(socioeconomic,
                                crs = st_crs(buildings_affected_blackout))
} else {
  message("socioeconomic CRS already matches buildings_affected_blackout CRS.")
}
```

```
Warning: socioeconomic CRS does not match. Transforming
          to match buildings_affected_blackout CRS
```

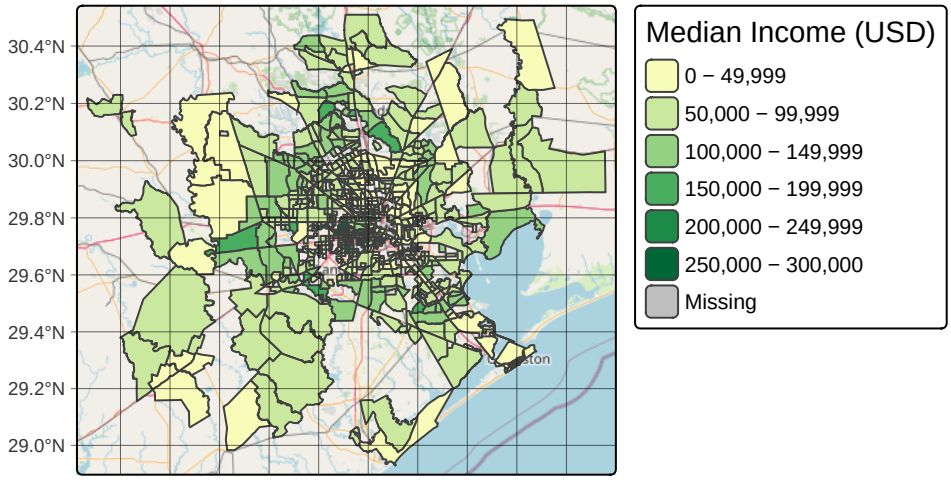
```
# Obtain intersecting features of socioeconomic
# and buildings_affected_blackout to identify affected census tracts
acs_affected <- st_filter(socioeconomic,
```

```
                        buildings_affected_blackout,
                        .predicate = st_intersects)
```

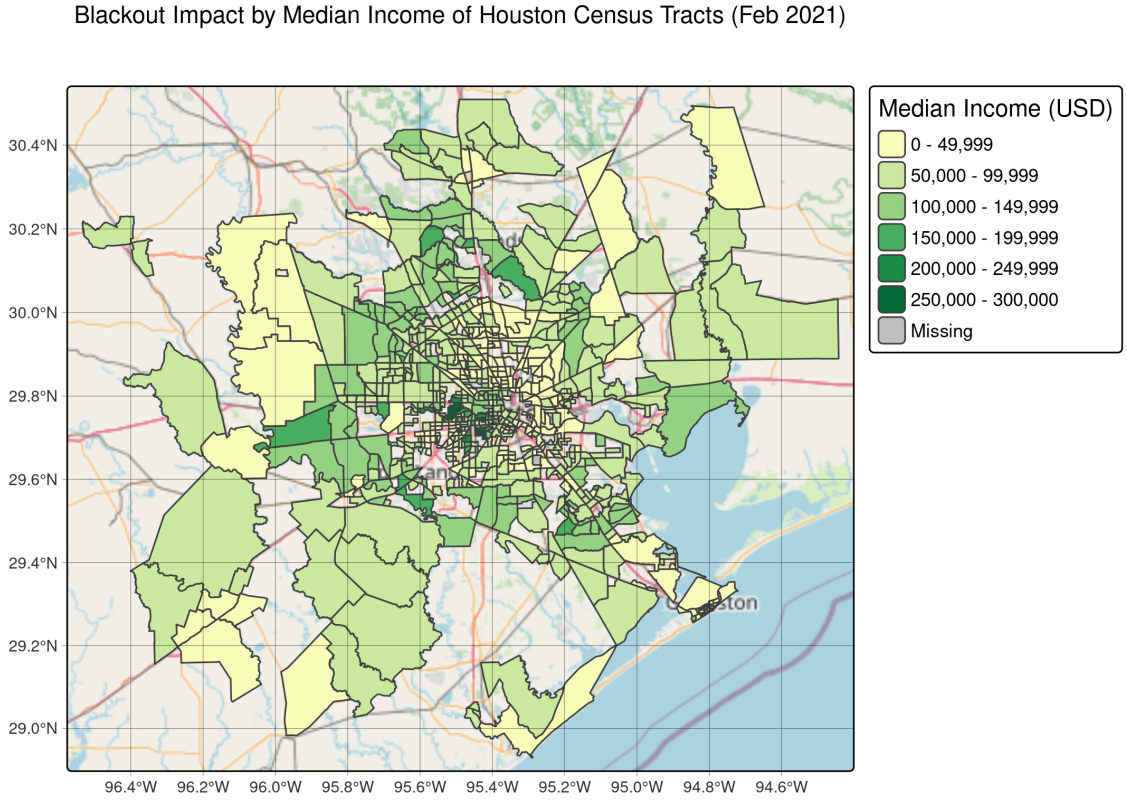
19. Create a map showcasing the median income (`B19013e1`, as defined in the metadata) in census tracts affected by the storm blackout.

```
# Create custom color palette
income_palette <- c("#f7fcb9", # Low income
                    "#addd8e",
                    "#31a354",
                    "#006837") # High income

# Create map for affected census tracts
acs_affected_map <- tm_shape(acs_affected) +

tm_polygons(
  col = "B19013e1", # Median income column
  title = "Median Income (USD)",
  palette = income_palette)+

  tm_layout(
    main.title = "Blackout Impact by Median Income of Houston Census Tracts (Feb 2021)",
    main.title.size = 2,
    main.title.position = "center",
    legend.outside = TRUE,
  ) +
   tm_graticules( # Add graticules to establish latitude and longitude network
    col = "black",
    lwd = 0.3, # Establish "thickness" of lines
    alpha = 0.6 ) +
  tm_basemap(server = "OpenStreetMap")

# Display census tract map
acs_affected_map
```

Blackout Impact by Median Income of Houston Census Tracts (Feb 2021)




```
# Save finalized map to figs
tmap_save(acs_affected_map,
          "figs/Affected_Census_Tracts.png")
```



Map

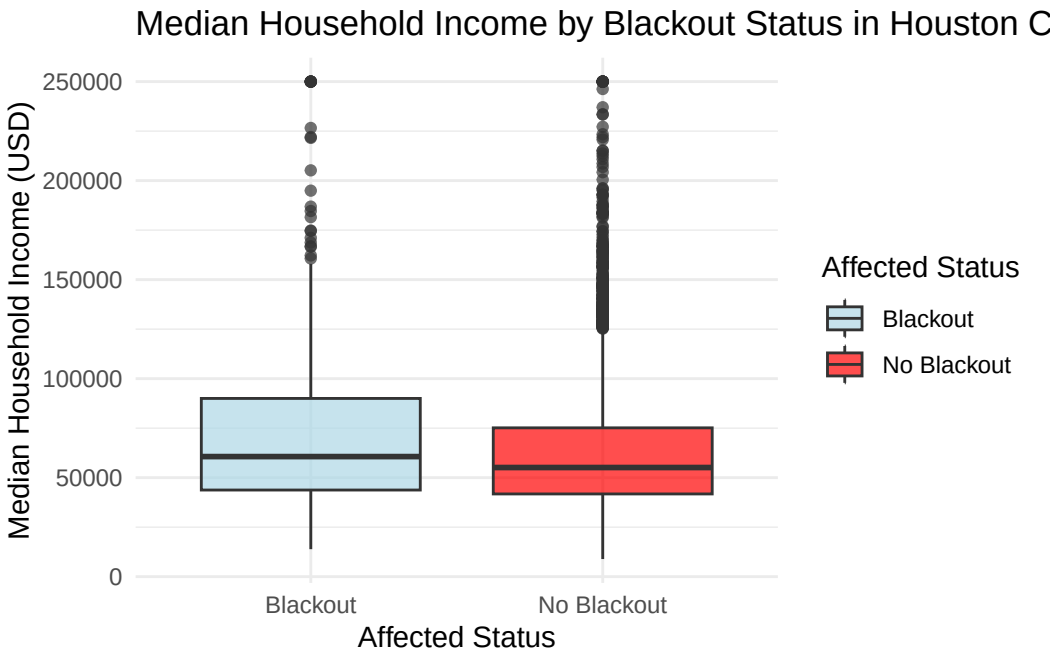
4. Finalized Version of affected census tracts based on median income in Houston.

19. Create a new dataframe that labels each census tract as either affected or unaffected by the blackout (`acs_compare`). Then, generate a boxplot and summary table to compare the distribution of median income between the two groups.

```
# Create a new column in acs_affected for affected census tracts
acs_affected$affected <- "Blackout" # Assign "blackout" status

# Grab unaffected census tracts from original socioeconomic variable
acs_unaffected <- socioeconomic[
  !(socioeconomic$GEOID %in% acs_affected$GEOID),] # Grab non-shared observations
```

```r
# Assign "No Blackout" status to
acs_unaffected$affected <- "No Blackout"

# Combine status
acs_compare <- rbind(acs_affected, acs_unaffected)

# Filter out missing values
acs_compare <- acs_compare %>% filter(!is.na(B19013e1))
```

```r
# Plot boxplot
acs_compare_plot <- acs_compare %>%
  ggplot(aes(x = affected, # Blackout status
             y = B19013e1, # Median income
             fill = affected)) + # Fill by blackout status
  geom_boxplot(alpha = 0.7) + # Adjust transparency)

  # Assign colors manually to status
  scale_fill_manual(values =
                      c("Blackout" = "lightblue",
                        "No Blackout" = "red")) +
  labs(
    title = "Median Household Income by Blackout Status in Houston Census Tracts",
    x = "Affected Status",
    y = "Median Household Income (USD)",
    fill = "Affected Status"
  ) +
  theme_minimal() # Set theme

acs_compare_plot
```




```
# Save finalized figure to figs
ggsave(
  filename = "figs/Affected_Census_Tracts_Boxplot.png",
  plot = acs_compare_plot,
  width = 8,
  height = 6)
```

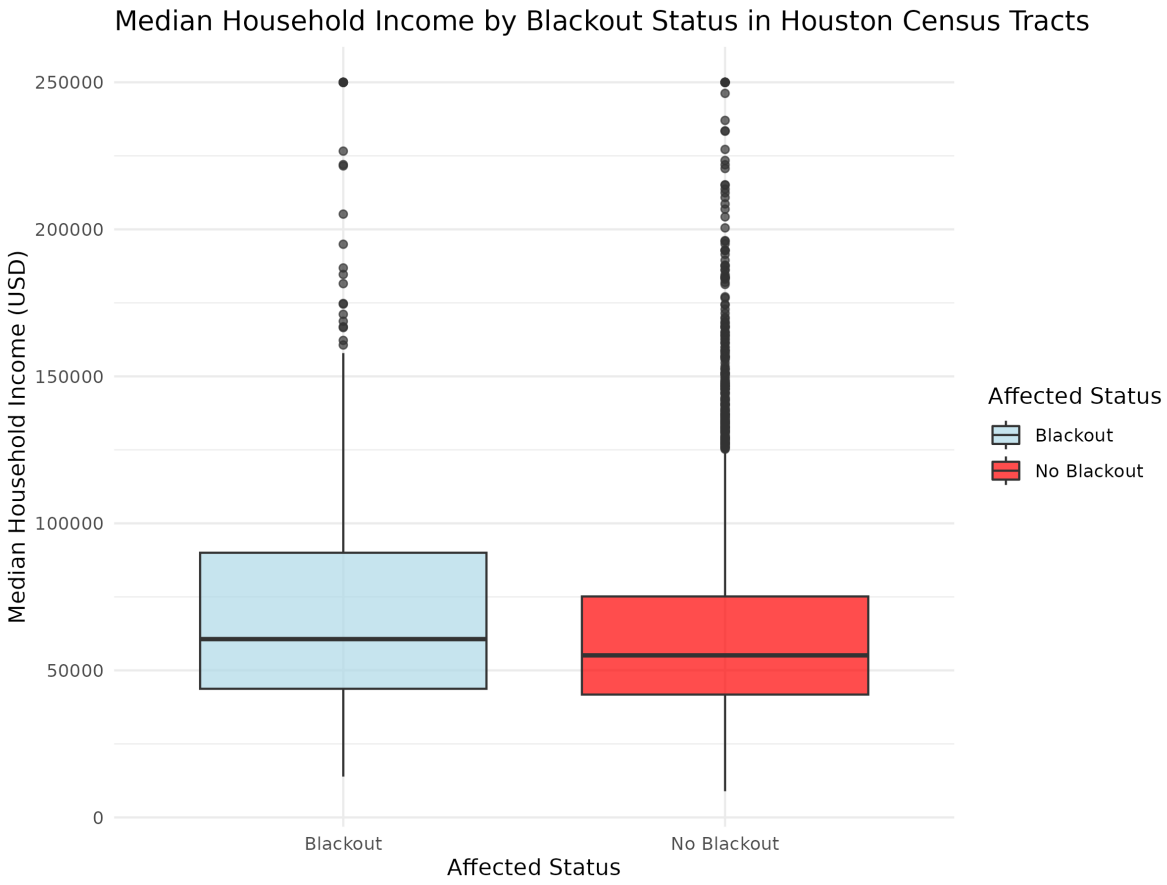


Figure 1. Finalized Version of median income distribution in affected Houston census tracts.

```r
# Summarize median income and n per affected group
income_summary <- acs_compare %>%
  st_drop_geometry() %>% # Drop geometries
  group_by(affected) %>%
  summarise(
    median_income = median(B19013e1, na.rm = T),
    num_tracts = n()
  )
```

```r
# Create summary table
income_summary %>%
  kbl( # Add title
    caption = "Summary of Median Income and Number of Census Tracts by Affected Status",
```

Table 1: Summary of Median Income and Number of Census Tracts by Affected Status

Affected Status	Median of Median Census Tract Income (USD)	Number of Census Tracts
Blackout	60642	708
No Blackout	55114	4494

```
    col.names = c("Affected Status", # Rename columns
                  "Median of Median Census Tract Income (USD)",
                  "Number of Census Tracts")) %>% kable_styling(
    full_width = FALSE, # Keep width narrow
    bootstrap_options = c("striped", "hover") # Format
  )
```

Reflection

Approximately 139,148 homes were affected by the blackout caused by Winter Storm Uri. Most of the impacted homes are located north of Sugar Land and along the city's outskirts, though numerous homes in the city center also experienced power loss. Coastal areas, including Galveston and Mustang Island, also saw clusters of affected homes. Interestingly, the city center had fewer power outages compared to the outskirts. When examined by median income, the affected census tracts are primarily in lower and middle income areas (~$0–$49,000 and $50,000–$99,999), whereas higher income households (over $100,000) are generally concentrated toward central Houston. On average, census tracts impacted by the blackout had a higher median income ($60,642) than those that were not affected ($55,114).

Occupations with higher salaries, such as those in the technology sector, are likely concentrated in the city center. Consequently, households with higher paying jobs may have been less likely to experience power outages compared to those with lower paying occupations. Other demographic factors, including race, poverty levels, and job type, may also influence the income distribution observed in the affected census tracts. A potential limitation of this study is the proximity of certain areas to nearby electrical grids, which may have helped sustain power for some residents. Geographic factors could further influence this distribution; for example, coastal and rural areas may be more vulnerable to outages due to underdeveloped infrastructure.