Appendix A

Owino Raymond

Load Required Libraries

```
library(tidyverse)        # Data manipulation
library(ggplot2)      # Plotting
library(viridis)      # Color scales for plots
library(sf)           # Simple Features for spatial data
library(spmodel)      # Spatial linear model (for spatial prediction)
library(gstat)        # For semivariogram fitting and spatial analysis
library(tidyr)        # For reshaping and cleaning data (if necessary)
library(tigris)       # To load U.S. Census data
library(patchwork)
library(AICcmodavg)
```

Data Loading and Filtering

```
# Load the data
load("data/wfigs_az_sf_EPSG32612.RData")

# convert pop.den as numeric
wfigs_az_sf$pop.density <- as.numeric(wfigs_az_sf$pop.density)

# Clean and select relevant columns, then rename for clarity
fire_size <- wfigs_az_sf %>%
  select(OBJECTID, FireDiscoveryDateTime, IncidentSize, FireCause,
         IncidentTypeCategory, POOCounty, x, y, tmax, tmin, prcp, mean_slope,
         mean_grass, mean_forest, mean_shrub, Temp_Max_Buffered,
         Temp_Min_Buffered, Precipitation_Buffered, Elevation,
         pop.density, pop., distance_rd_primary, distance_rd_min_all,
         distance_rd_secondary, distance_rd_4wd) %>%
  rename(
    ID = OBJECTID,
    Date = FireDiscoveryDateTime,
    Size = IncidentSize,
    Cause = FireCause,
    Category = IncidentTypeCategory,
    County = POOCounty,
    Long = x,
    Lat = y,
    Max_day_temp = tmax,
    Min_day_temp = tmin,
    Prcp = prcp,
    pSlope = mean_slope,
```

```
    Grass_p = mean_grass,
    Forest_p = mean_forest,
    Shrub_p = mean_shrub,
    Max_ann_temp = Temp_Max_Buffered,
    Min_ann_temp = Temp_Min_Buffered,
    Prcp_ann = Precipitation_Buffered,
    Elevation = Elevation,
    pop_density = pop.density,
    Population = pop.,
    Pri_rd = distance_rd_primary,
    All_rd = distance_rd_min_all,
    Sec_rd = distance_rd_secondary,
    Dist_4WD = distance_rd_4wd
  )

# Filter for wildfires and focus on human or natural causes
wild_fires <- fire_size %>%
  filter(Category == "WF") %>%
  filter(Cause == "Human" | Cause == "Natural")

# Filter fires in Coconino county with a size greater than or equal to 50 acres
coconino <- wild_fires %>%
  filter(County == "Coconino") %>%
  filter(Size >= 1000)
```

Visualize data in relation to Arizona

```
# Get Arizona state outline
az_outline <- states(cb = TRUE) %>%
  filter(NAME == "Arizona") %>%
  st_transform(crs = 26912)
```

```
##   |                                                                      |
```

```
# Get Coconino County
coconino_county <- counties("AZ", cb = TRUE) %>%
  filter(NAME == "Coconino") %>%
  st_transform(crs = 26912)
```

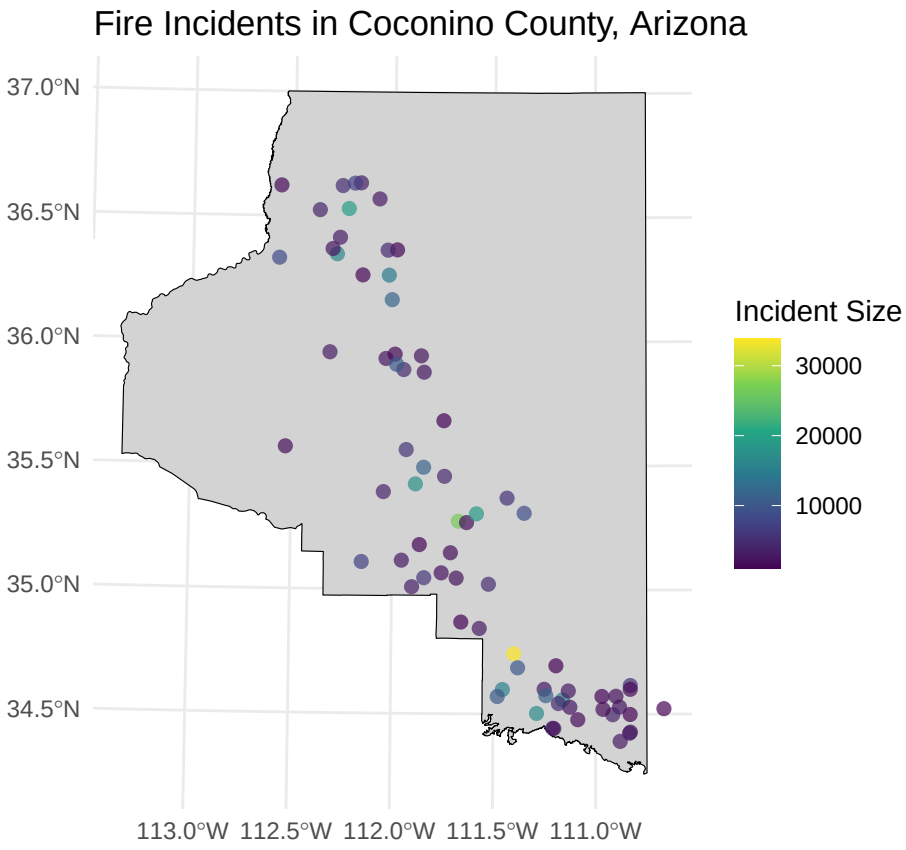
```
##   |                                                                      |
```

```
# Load Census Tracts for Coconino County (2020 data)
census_tracts_sf <- tracts(state = "AZ", county = "Coconino", year = 2020, cb = TRUE) %>%
  st_transform(crs = 26912)
```

```
##   |                                                                      |
```

```
# Assuming coconino_sf contains the fire incident points
# Transform to the same CRS if needed
coconino_sf <- st_transform(coconino, crs = 26912)

# Plot
original_plot <- ggplot() +
  geom_sf(data = coconino_county, fill = "lightgray", color = "black", size = 1) +
  geom_sf(data = coconino_sf, aes(color = Size), size = 2, alpha = 0.7) +
```

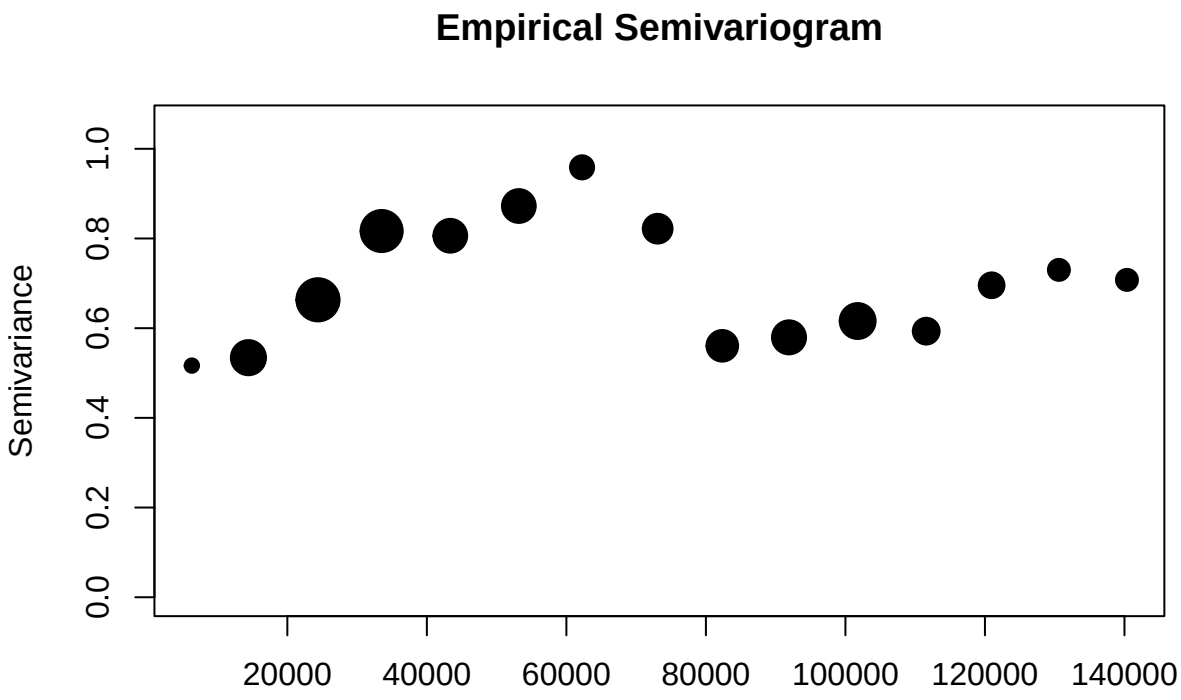
```
  scale_color_viridis_c(option = "YlOrRd") +
  labs(title = "Fire Incidents in Coconino County, Arizona", color = "Incident Size") +
  theme_minimal()

original_plot
```




```
# save
# ggsave("coconino_fire_map.png", width = 10, height = 8, dpi = 300)
```

Semivariogram

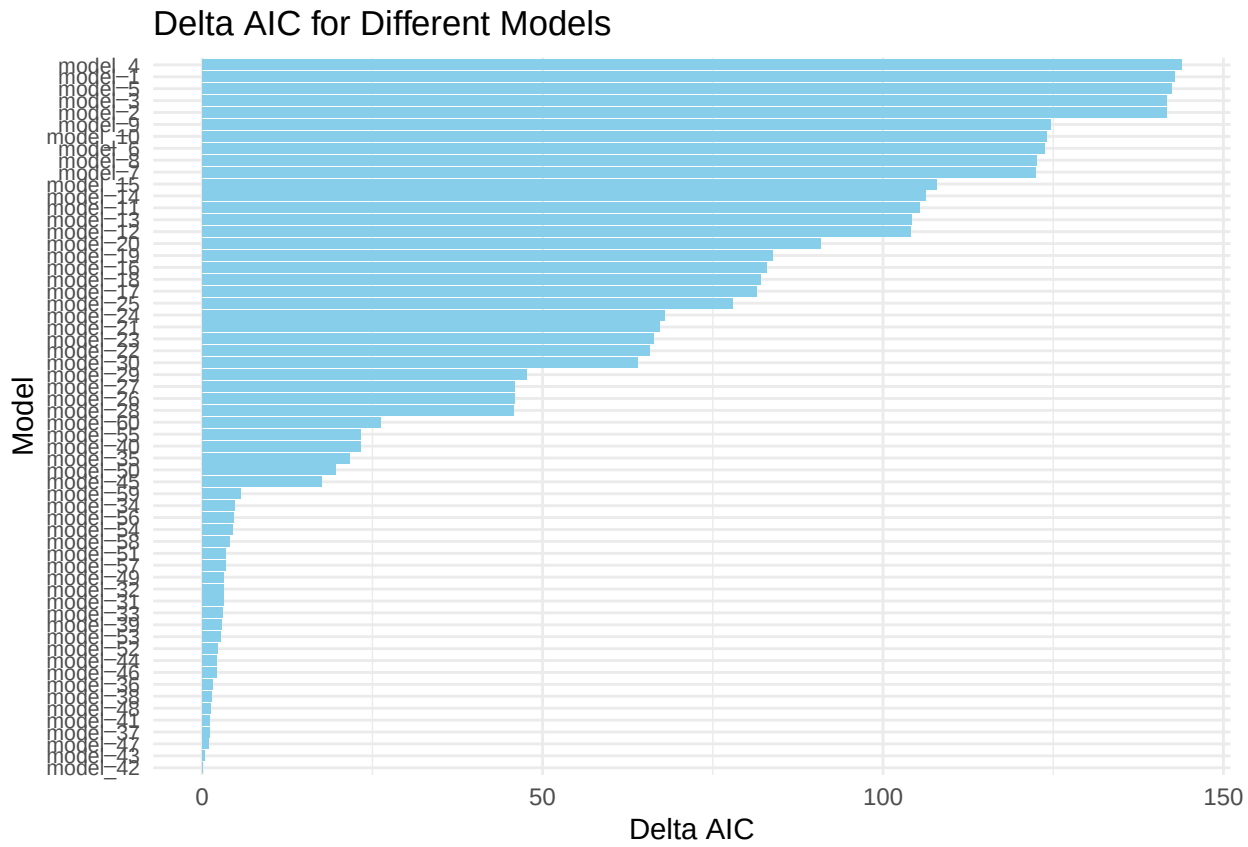
```
# Fit the model
fire_esv <- esv(
  log(Size) ~ pSlope + Grass_p + Forest_p + Max_ann_temp + Min_ann_temp + Prcp_ann + pop_density,
  data = coconino_sf
)


# Plot the semivariogram to assess spatial autocorrelation
plot(fire_esv)
```



Select best linear model

```
# List of all variables
variables <- c("pSlope", "Grass_p", "Forest_p", "Max_ann_temp", "Min_ann_temp",
               "Prcp_ann", "pop_density","Population", "Pri_rd", "All_rd", "Sec_rd", "Dist_4WD")

# List of spatial covariance types
spcov_types <- c("exponential", "gaussian", "spherical", "matern","none")

# Initialize an empty list to store models
models <- list()

# Counter for model names
model_counter <- 1

# Loop through variables
for (i in 1:length(variables)) {
  # Create formula with current set of variables and log-transformed Size
  formula <- as.formula(paste("Size ~", paste(variables[1:i], collapse = " + ")))

  # Loop through spatial covariance types
  for (spcov in spcov_types) {
    # Fit model
    model <- try(splm(formula, data = coconino_sf, spcov_type = spcov), silent = TRUE)
```

```
    # If model fitting was successful, add to list
    if (!inherits(model, "try-error")) {
      model_name <- paste0("model_", model_counter)
      models[[model_name]] <- model
      model_counter <- model_counter + 1
    }
  }
}

# Calculate AIC for all models
aic_values <- sapply(models, AIC)

# Find the model with the lowest AIC
best_model <- models[[which.min(aic_values)]]

# Calculate delta AICs
min_aic <- min(aic_values)
delta_aic <- aic_values - min_aic

# Create a data frame for plotting
plot_data <- data.frame(
  Model = names(delta_aic),
  DeltaAIC = delta_aic
)

# Sort the data frame by DeltaAIC
plot_data <- plot_data[order(plot_data$DeltaAIC), ]

# Create the plot
ggplot(plot_data, aes(x = reorder(Model, DeltaAIC), y = DeltaAIC)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +
  labs(title = "Delta AIC for Different Models",
       x = "Model",
       y = "Delta AIC") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 8))
```




```
# Print the best model
cat("Best model:\n")
```

```
## Best model:
```

```
print(summary(best_model))
```

```
##
## Call:
## splm(formula = formula, data = coconino_sf, spcov_type = spcov)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -13436  -3485   -939   1843  21910
##
## Coefficients (fixed):
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   9.253e+04  2.387e+04   3.876 0.000106 ***
## pSlope        1.183e+01  8.533e+01   0.139 0.889706
## Grass_p       1.186e+03  4.087e+03   0.290 0.771747
## Forest_p     -5.391e+03  3.178e+03  -1.696 0.089793 .
## Max_ann_temp -4.064e+03  1.242e+03  -3.272 0.001067 **
## Min_ann_temp  1.654e+03  1.140e+03   1.450 0.147039
## Prcp_ann     -7.421e+03  2.749e+03  -2.699 0.006953 **
## pop_density  -3.286e+08  7.405e+08  -0.444 0.657203
## Population    3.877e-03  6.232e-01   0.006 0.995037
## Pri_rd       -1.066e-01  3.604e-02  -2.957 0.003110 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Pseudo R-squared: 0.2678
##
## Coefficients (gaussian spatial covariance):
##       de       ie    range
##  5790546 29326176    32436
```

```
# Print the formula of the best model
cat("\nBest model formula:\n")
```

```
##
## Best model formula:
```

```
print(formula(best_model))
```

```
## Size ~ pSlope + Grass_p + Forest_p + Max_ann_temp + Min_ann_temp +
##     Prcp_ann + pop_density + Population + Pri_rd
```

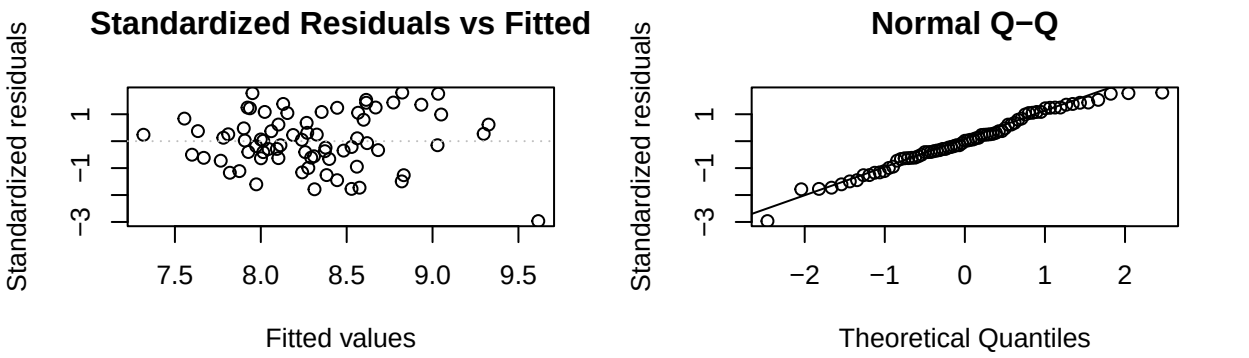
Fit spatial linear model

```
# Spatial linear model
spmod <- splm(log(Size) ~ pSlope + Grass_p + Forest_p + Max_ann_temp + Min_ann_temp + Prcp_ann + pop_de

# Display summary and diagnostics of the fitted spatial model
summary(spmod)
```

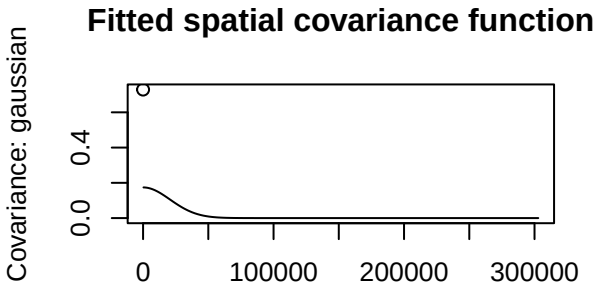
```
##
## Call:
## splm(formula = log(Size) ~ pSlope + Grass_p + Forest_p + Max_ann_temp +
##     Min_ann_temp + Prcp_ann + pop_density + Population + Pri_rd,
##     data = coconino_sf, spcov_type = "gaussian")
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0343 -0.4827 -0.1032  0.5320  1.6863
##
## Coefficients (fixed):
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.826e+01  3.389e+00   5.387 7.16e-08 ***
## pSlope        9.238e-03  1.223e-02   0.755   0.4500
## Grass_p       2.773e-01  5.822e-01   0.476   0.6338
## Forest_p     -5.344e-01  4.479e-01  -1.193   0.2328
## Max_ann_temp -5.156e-01  1.771e-01  -2.911   0.0036 **
## Min_ann_temp  2.264e-01  1.648e-01   1.374   0.1695
## Prcp_ann     -8.095e-01  3.881e-01  -2.086   0.0370 *
## pop_density   1.640e+04  1.083e+05   0.151   0.8796
## Population    6.920e-05  9.247e-05   0.748   0.4542
## Pri_rd       -1.013e-05  5.304e-06  -1.911   0.0561 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Pseudo R-squared: 0.2248
##
```

```
## Coefficients (gaussian spatial covariance):
##        de        ie     range
##     0.174     0.555 29119.613
```

```
par(mfrow = c(2, 2))
plot(spmod)
par(mfrow = c(1, 1))
```



) ~ pSlope + Grass_p + Forest_p + Max_ann_temp +) ~ pSlope + Grass_p + Forest_p + Max_ann_temp +



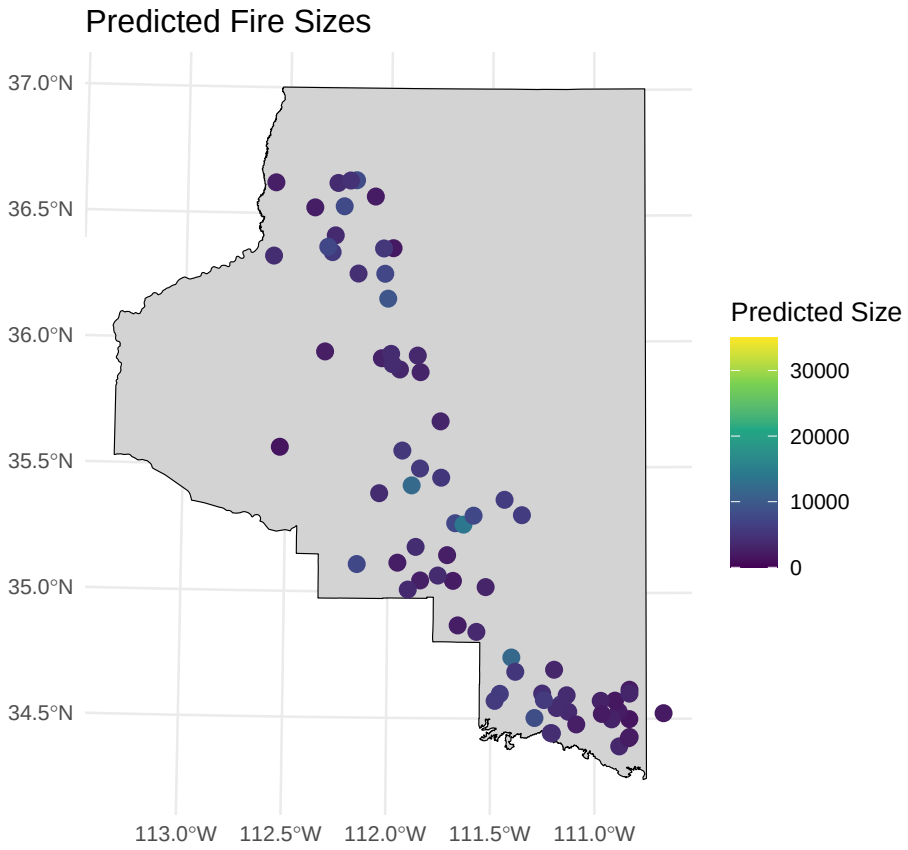
) ~ pSlope + Grass_p + Forest_p + Max_ann_temp +

Prediction from the model

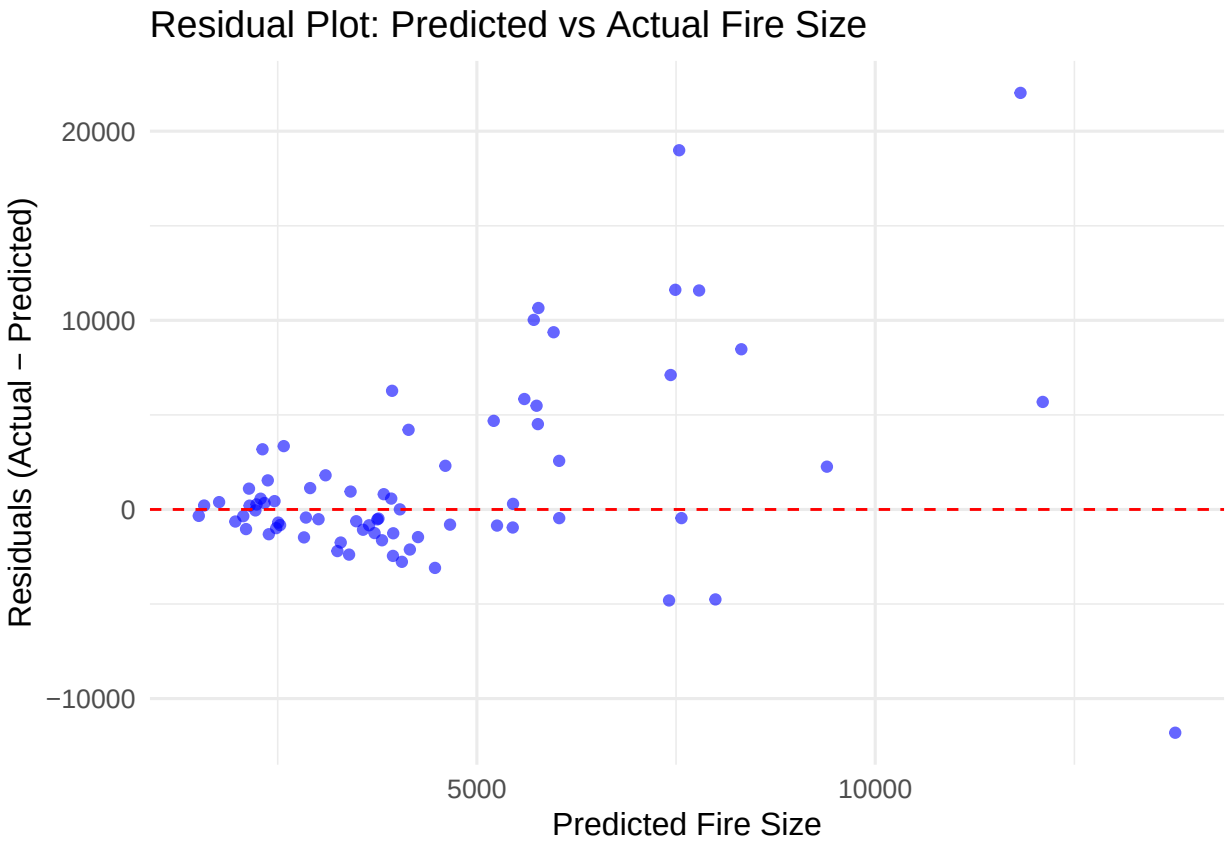
```
# Make predictions
predictions_log_actual <- predict(spmod, newdata = coconino_sf, type = "response")

# Convert log predictions back to the original scale (Size)
coconino_sf$predicted_size <- exp(predictions_log_actual)

# Plot the predicted fire sizes on the map
plot_actual <- ggplot() +
  geom_sf(data = coconino_county, fill = "lightgray", color = "black", size = 1) +  # Coconino County b
  geom_sf(data = coconino_sf, aes(color = predicted_size), size = 2.5) +
  scale_color_viridis_c(option = "YlOrRd", limits = c(0, 35000))+
  labs(title = "Predicted Fire Sizes", color = "Predicted Size") +
  theme_minimal(base_size = 10)
```

```
## Warning in viridisLite::viridis(n, alpha, begin, end, direction, option):
## Option 'YlOrRd' does not exist. Defaulting to 'viridis'.
```

```r
# Show the predicted fire sizes plot
print(plot_actual)
```



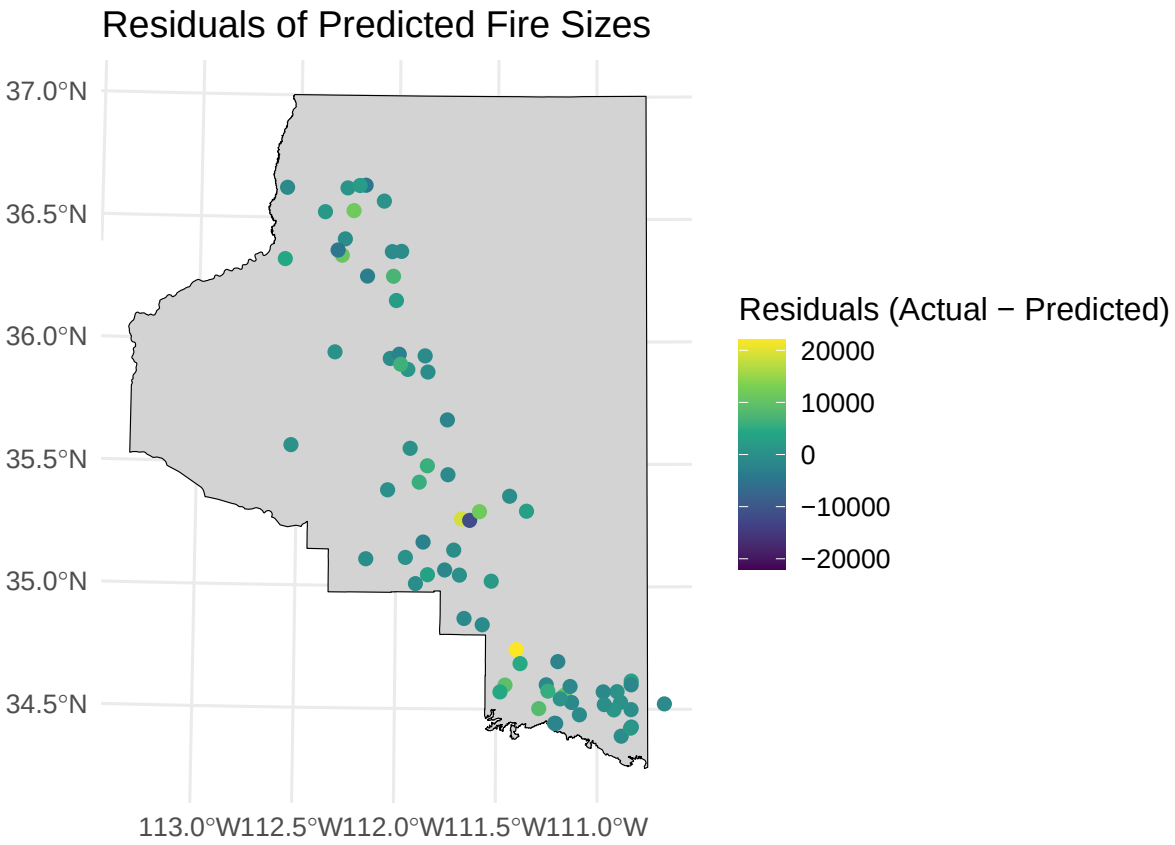

```r
# Calculate residuals: Actual fire size - Predicted fire size
coconino_sf$residuals <- coconino_sf$Size - coconino_sf$predicted_size

# Create a residual plot (Predicted vs Residuals)
residual_plot <- ggplot(coconino_sf, aes(x = predicted_size, y = residuals)) +
  geom_point(color = "blue", alpha = 0.6) +  # Scatter plot of residuals
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residual Plot: Predicted vs Actual Fire Size",
       x = "Predicted Fire Size", y = "Residuals (Actual - Predicted)") +
  theme_minimal(base_size = 12)  # Minimal theme for the plot

# Show the residual plot
print(residual_plot)
```



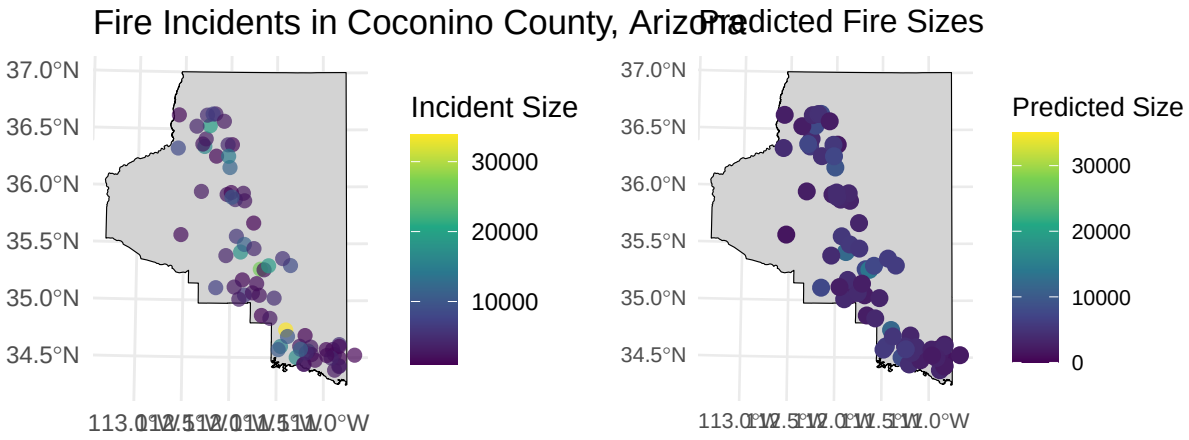

```
# Plot the residuals on the map
residual_map <- ggplot() +
  geom_sf(data = coconino_county, fill = "lightgray", color = "black", size = 1)+
  geom_sf(data = coconino_sf, aes(color = residuals), size = 2) +
  scale_color_viridis_c(option = "YlOrRd", limits = c(-max(abs(coconino_sf$residuals)), max(abs(coconin
  labs(title = "Residuals of Predicted Fire Sizes", color = "Residuals (Actual - Predicted)") +
  theme_minimal(base_size = 12)
```

```
## Warning in viridisLite::viridis(n, alpha, begin, end, direction, option):
## Option 'YlOrRd' does not exist. Defaulting to 'viridis'.
```

```
# Show the residual map
print(residual_map)
```




```
# compare original and predicted
original_plot + plot_actual
```


Fire Incidents in Coconino County, Arizona
37.0°N
36.5°N
36.0°N
35.5°N
35.0°N
34.5°N
113.0°W
Incident Size
30000
20000
10000
Predicted Fire Sizes
37.0°N
36.5°N
36.0°N
35.5°N
35.0°N
34.5°N
113.0°W
Predicted Size
30000
20000
10000
0

Appendix B

Matthew Wallace

Libraries

```
library(dplyr)
library(tigris)
library(sf)
library(ggplot2)
library(tidycensus)
library(spatstat)
library(spmodel)
library(cowplot)
library(terra)
library(stars)
```

Data Loading

```
# Arizona
load("wfigs_az_sf_EPSG32612.RData")
st_crs(wfigs_az_sf) <- 32612
az_fires_rx <- wfigs_az_sf %>% filter(IncidentTypeCategory=="RX")
az_fires_wf <- wfigs_az_sf %>% filter(IncidentTypeCategory=="WF")

az_fires_wf_hum <- az_fires_wf %>% filter(FireCause=="Human")
az_fires_wf_nat <- az_fires_wf %>% filter(FireCause=="Natural")
az_fires_wf_un <- az_fires_wf %>% filter(FireCause=="Undetermined" |
                                          FireCause=="Unknown")

# Coconino
coconino_sf <- counties(state = "AZ") %>%
  filter(NAME == "Coconino")
```

```
##   |                                                                      |
```

```
coconino_sf <- st_transform(coconino_sf, 32612)
wfigs_coco_sf <- wfigs_az_sf %>%
  st_filter(coconino_sf, .predicate = st_within)

coco_fires_rx <- wfigs_coco_sf %>% filter(IncidentTypeCategory=="RX")
coco_fires_wf <- wfigs_coco_sf %>% filter(IncidentTypeCategory=="WF")

coco_fires_wf_hum <- coco_fires_wf %>% filter(FireCause=="Human")
coco_fires_wf_nat <- coco_fires_wf %>% filter(FireCause=="Natural")
```

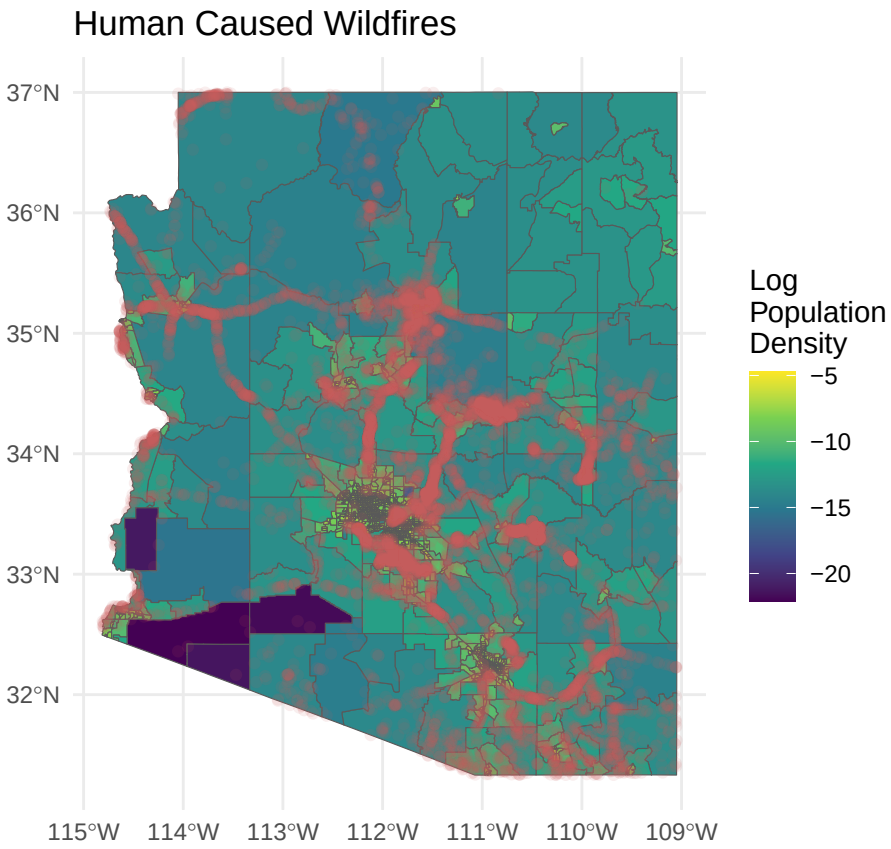
```
coco_fires_wf_un <- coco_fires_wf %>% filter(FireCause=="Undetermined" |
                                              FireCause=="Unknown")

coco_fires_wf_hum_lg <- coco_fires_wf_hum %>% filter(IncidentSize >= 1000)
coco_fires_wf_hum_sm <- coco_fires_wf_hum %>% filter(IncidentSize < 1000)
coco_fires_wf_nat_lg <- coco_fires_wf_nat %>% filter(IncidentSize >= 1000)
coco_fires_wf_nat_sm <- coco_fires_wf_nat %>% filter(IncidentSize < 1000)

coconino.bbox <- st_bbox(coconino_sf)
coco_pred.owin <- as.owin(coconino_sf)
```

Population Density Exploration

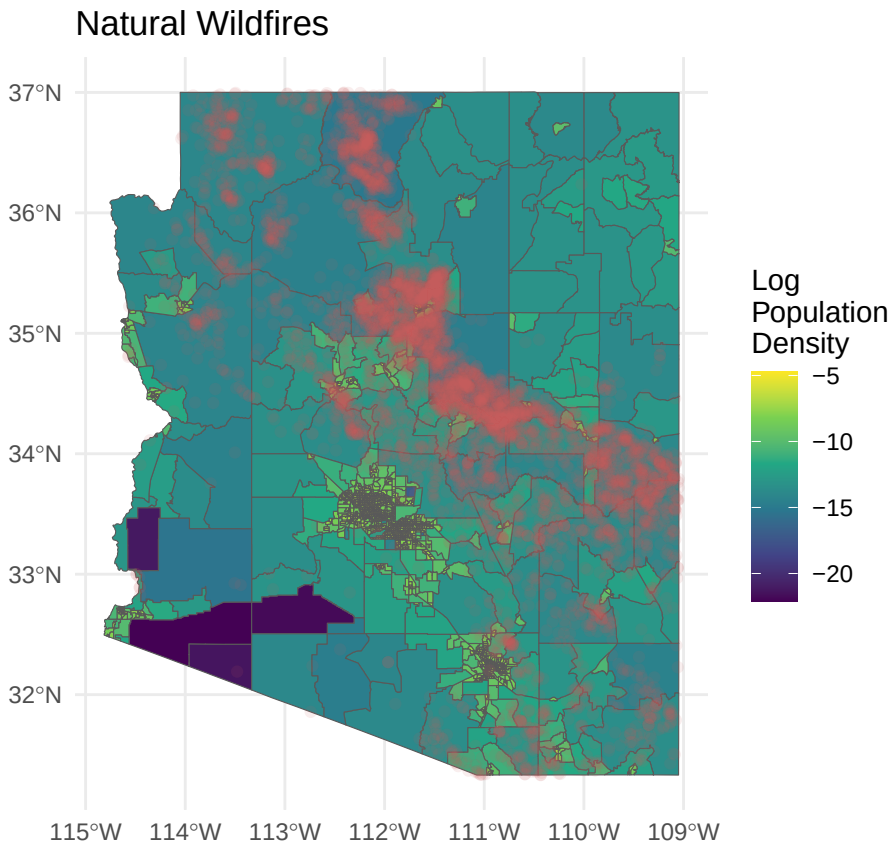
```
population_data <- get_decennial(
  geography = "tract",
  variables = c(population="P001001"),
  state = "AZ",
  year = 2010,
  geometry = TRUE
)
```

```
##   |                                                                              |
```

```
population_data$value[population_data$value==0] <- 1
population_data$variable <-  population_data$value/st_area(population_data$geometry)
names(population_data) <- c("GEOID","NAME","pop.density","pop.","geometry")

ggplot() +
  geom_sf(aes(fill=as.numeric(log(pop.density))), data=population_data) +
  geom_sf(data=az_fires_wf_hum, alpha=0.1, col="indianred") +
  scale_fill_viridis_c() +
  labs(fill = "Log\nPopulation\nDensity", title="Human Caused Wildfires") +
  theme_minimal()
```




```
ggplot() +
  geom_sf(aes(fill=as.numeric(log(pop.density))), data=population_data) +
  geom_sf(data=az_fires_wf_nat, alpha=0.1, col="indianred") +
  scale_fill_viridis_c() +
  labs(fill = "Log\nPopulation\nDensity", title="Natural Wildfires") +
  theme_minimal()
```

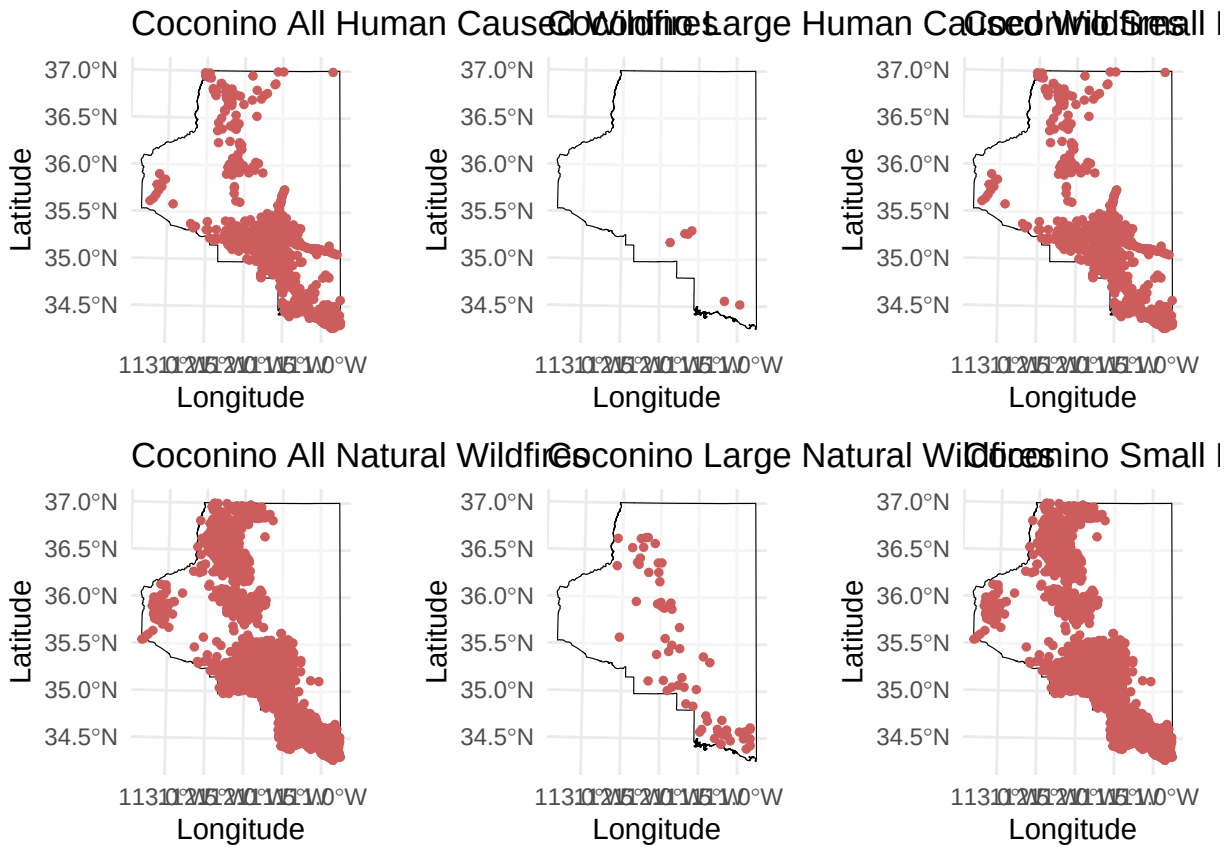


Prediction Grid Data

Response

```
# Human Caused PPP
coco_wf.ppp <- as.ppp(coco_fires_wf, W=coco_pred.owin)
coco_wf_hum.ppp <- as.ppp(coco_fires_wf_hum, W=coco_pred.owin)
coco_wf_hum_lg.ppp <- as.ppp(coco_fires_wf_hum_lg, W=coco_pred.owin)
coco_wf_hum_sm.ppp <- as.ppp(coco_fires_wf_hum_sm, W=coco_pred.owin)

# Natural PPP
coco_wf_nat.ppp <- as.ppp(coco_fires_wf_nat, W=coco_pred.owin)
coco_wf_nat_lg.ppp <- as.ppp(coco_fires_wf_nat_lg, W=coco_pred.owin)
coco_wf_nat_sm.ppp <- as.ppp(coco_fires_wf_nat_sm, W=coco_pred.owin)

# Human Caused Data Frames
coco_wf_hum.df <- data.frame(x = coco_wf_hum.ppp$x, y = coco_wf_hum.ppp$y)
coco_wf_hum_lg.df <- data.frame(x = coco_wf_hum_lg.ppp$x, y = coco_wf_hum_lg.ppp$y)
coco_wf_hum_sm.df <- data.frame(x = coco_wf_hum_sm.ppp$x, y = coco_wf_hum_sm.ppp$y)

# Natural Data Frames
coco_wf_nat.df <- data.frame(x = coco_wf_nat.ppp$x, y = coco_wf_nat.ppp$y)
coco_wf_nat_lg.df <- data.frame(x = coco_wf_nat_lg.ppp$x, y = coco_wf_nat_lg.ppp$y)
coco_wf_nat_sm.df <- data.frame(x = coco_wf_nat_sm.ppp$x, y = coco_wf_nat_sm.ppp$y)
```

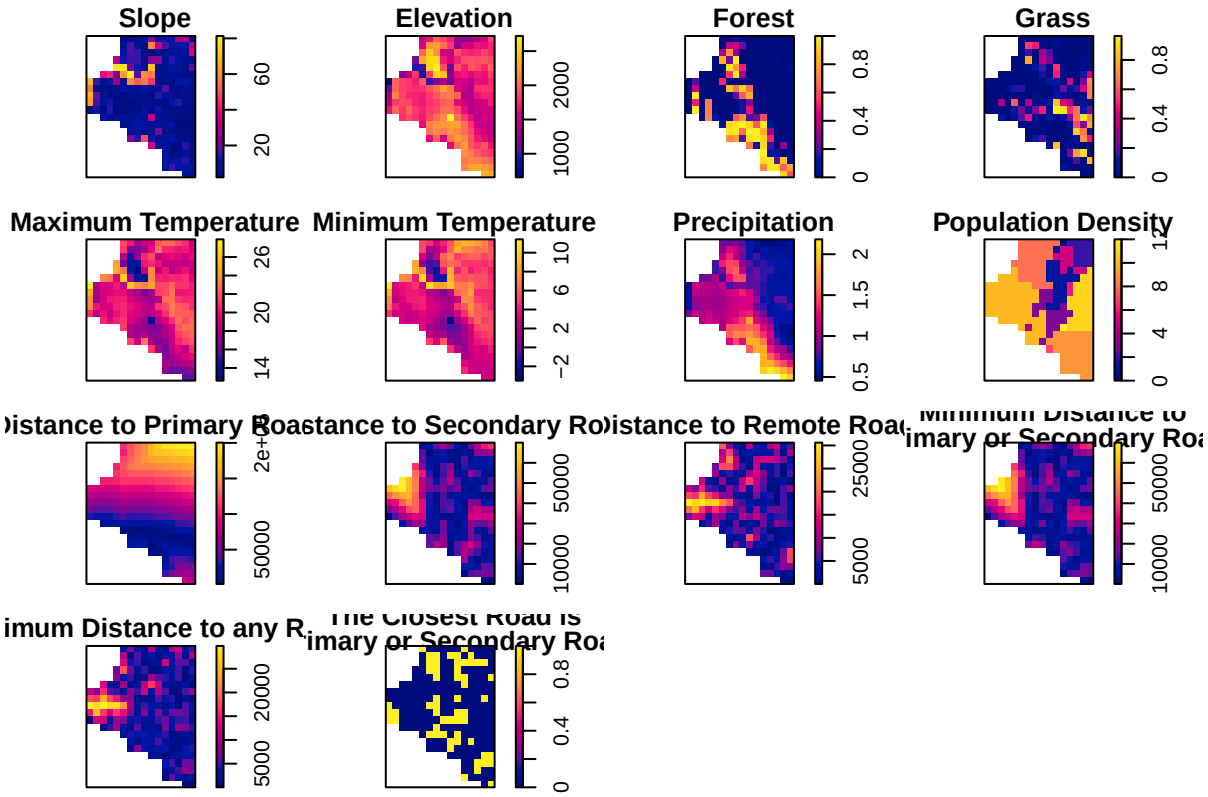
```r
# Create individual plots
hum_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_hum.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino All Human Caused Wildfires", x = "Longitude", y = "Latitude")

hum_lg_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_hum_lg.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino Large Human Caused Wildfires", x = "Longitude", y = "Latitude")

hum_sm_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_hum_sm.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino Small Human Caused Wildfires", x = "Longitude", y = "Latitude")

nat_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_nat.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino All Natural Wildfires", x = "Longitude", y = "Latitude")

nat_lg_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_nat_lg.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino Large Natural Wildfires", x = "Longitude", y = "Latitude")

nat_sm_plot <- ggplot() +
  geom_sf(data = coconino_sf, fill = "white", color = "black", alpha = 0.5) +
  geom_point(data = coco_wf_nat_sm.df, aes(x = x, y = y), color = "indianred", size = 1) +
  theme_minimal() +
  labs(title = "Coconino Small Natural Wildfires", x = "Longitude", y = "Latitude")

# Arrange in a 2x3 grid
plot_grid(hum_plot, hum_lg_plot, hum_sm_plot,
          nat_plot, nat_lg_plot, nat_sm_plot,
          nrow=2, ncol=3)
```



Predictors

```
load("az_prediction_grid_sf.RData")
get_im <- function(x){
  coconino.owin <- owin(yrange = c(coconino.bbox["xmin"], coconino.bbox["xmax"]),
                        xrange = c(-coconino.bbox["ymax"], -coconino.bbox["ymin"]))
  rast <- rast(extent = coconino.bbox, res = 15000)
  rasterized <- rasterize(az_prediction_grid_sf, rast, field=x)
  matrix <- t(as.matrix(rasterized, wide=TRUE))
  im <- rotate.im(as.im(matrix, W=coconino.owin), angle = -pi/2)
  return(im)
}
predictors <- c("mean_slope",
                "Elevation",
                "mean_forest",
                "mean_grass",
                "Temp_Max_Buffered",
                "Temp_Min_Buffered",
                "Precipitation_Buffered",
                "pop.density",
                "distance_rd_primary",
                "distance_rd_secondary",
                "distance_rd_4wd",
                "distance_rd_min_prisec",
                "distance_rd_min_all",
```

```
                "distance_rd_min_isprisec")

pred_titles <- c("Slope",
                 "Elevation",
                 "Forest",
                 "Grass",
                 "Maximum Temperature",
                 "Minimum Temperature",
                 "Precipitation",
                 "Population Density",
                 "Distance to Primary Road",
                 "Distance to Secondary Road",
                 "Distance to Remote Road",
                 "Minimum Distance to\nPrimary or Secondary Road",
                 "Minimum Distance to any Road",
                 "The Closest Road is\nPrimary or Secondary Road")
par(mfrow=c(4,4),
    mar = c(1, 1, 1, 1),
    oma = c(1, 1, 1, 1))
for (i in 1:length(predictors)){
  plot(get_im(predictors[i]), main=pred_titles[i])
}

predictors.im <- lapply(predictors, get_im)
names(predictors.im) <- predictors
```

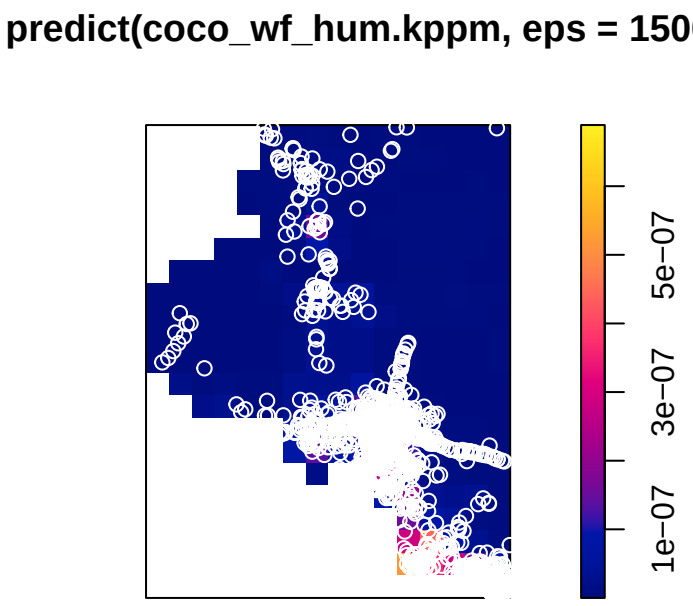


Models

Human Caused

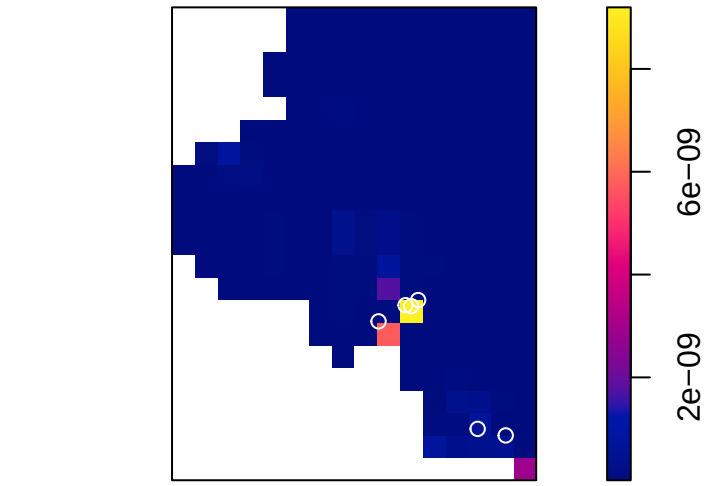
All Wildfires

```
# Fitting
coco_wf_hum.kppm <- kppm(unmark(coco_wf_hum.ppp)~., data=predictors.im,
                         clusters = "LGCP", model = "exponential")
#step(coco_wf_hum.kppm)

plot(predict(coco_wf_hum.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_hum), col="white")
```

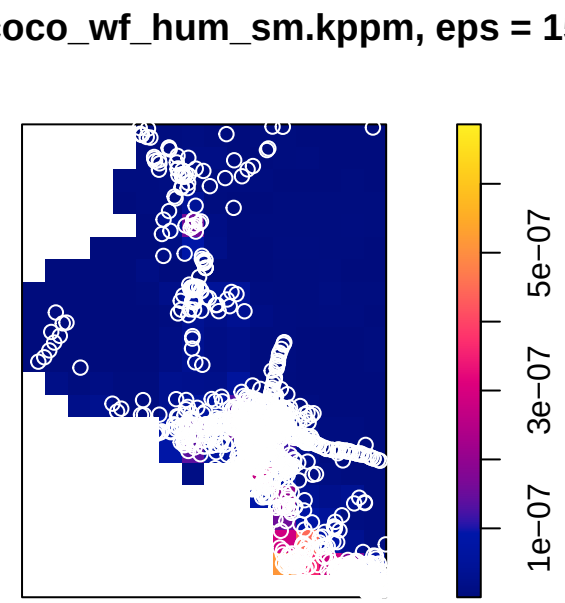


Thresholded Wildfires

```
# Fitting
coco_wf_hum_lg.kppm <- kppm(unmark(coco_wf_hum_lg.ppp)~., data=predictors.im,
                            clusters = "LGCP", model = "exponential")
plot(predict(coco_wf_hum_lg.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_hum_lg), col="white")
```



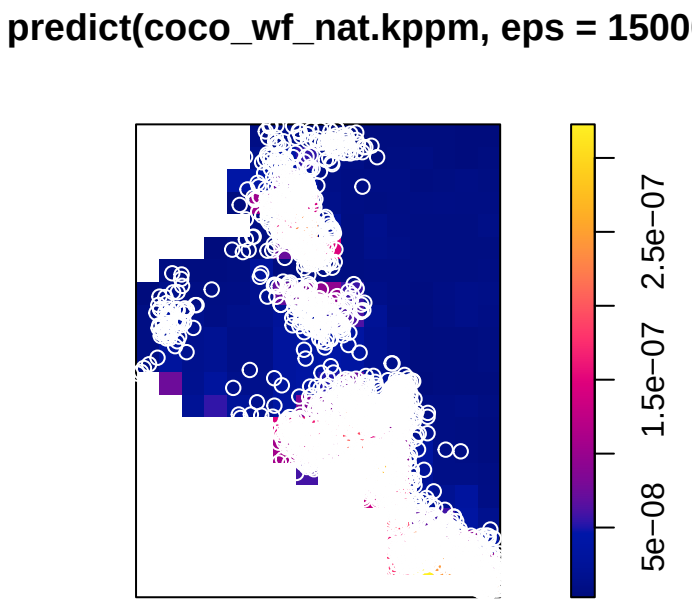

```
coco_wf_hum_sm.kppm <- kppm(unmark(coco_wf_hum_sm.ppp)~., data=predictors.im,
                            clusters = "LGCP", model = "exponential")
plot(predict(coco_wf_hum_sm.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_hum_sm), col="white")
```



Natural

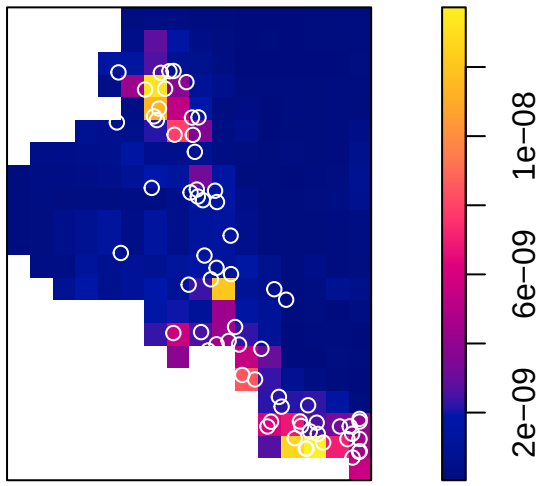
All Wildfires

```
# Fitting
coco_wf_nat.kppm <- kppm(unmark(coco_wf_nat.ppp)~., data=predictors.im,
                         clusters = "LGCP", model = "exponential")
#step(coco_wf_nat.kppm)

plot(predict(coco_wf_nat.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_nat), col="white")
```

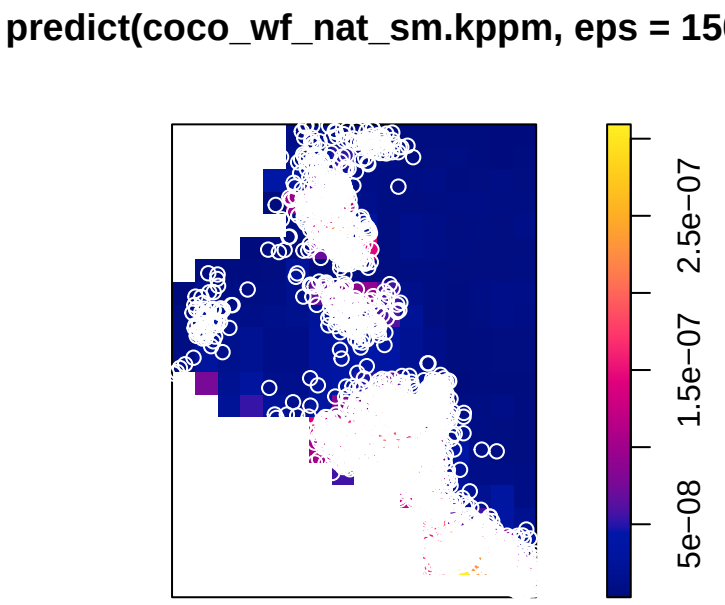


Threshold Wilfires

```
coco_wf_nat_lg.kppm <- kppm(unmark(coco_wf_nat_lg.ppp)~., data=predictors.im,
                            clusters = "LGCP", model = "exponential")
plot(predict(coco_wf_nat_lg.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_nat_lg), col="white")
```




```
coco_wf_nat_sm.kppm <- kppm(unmark(coco_wf_nat_sm.ppp)~., data=predictors.im,
                            clusters = "LGCP", model = "exponential")
plot(predict(coco_wf_nat_sm.kppm, eps = 15000))
points(st_coordinates(coco_fires_wf_nat_sm), col="white")
```



CSR Analysis

Setup

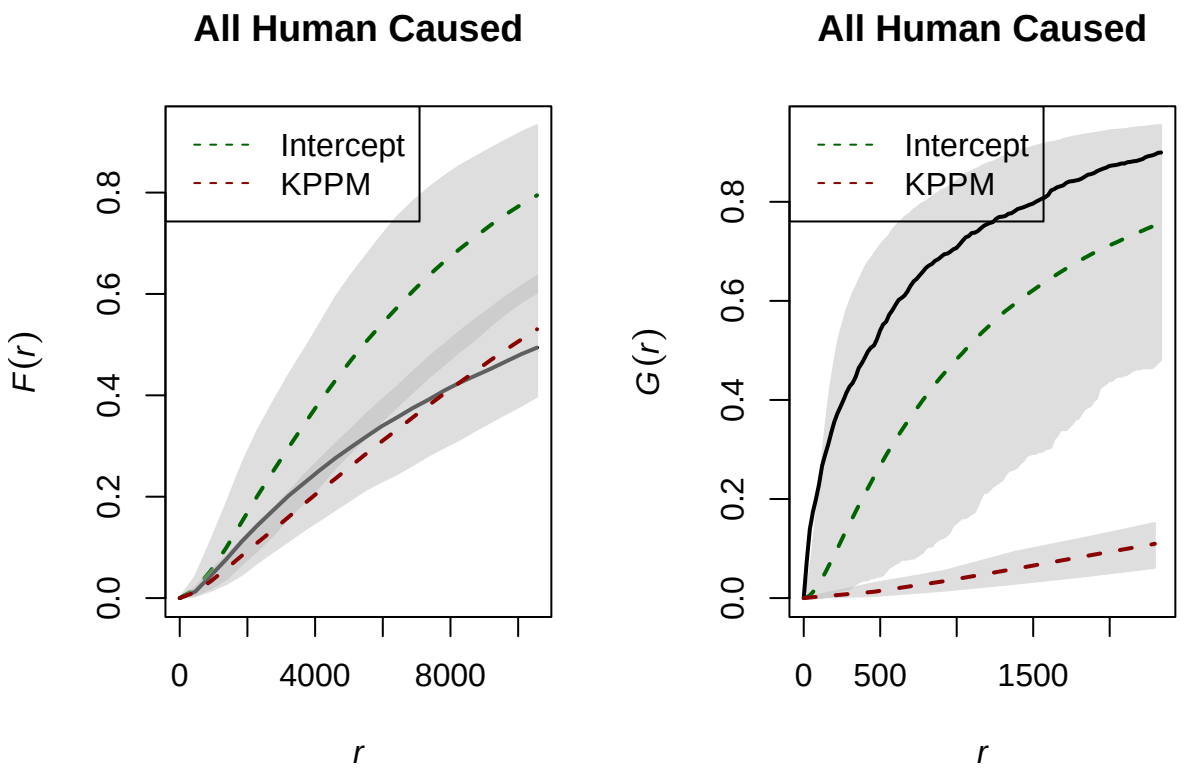
```
coco_wf_hum_naive.kppm <- kppm(unmark(coco_wf_hum.ppp)~1,
                               clusters = "LGCP", model = "exponential")
coco_wf_hum_lg_naive.kppm <- kppm(unmark(coco_wf_hum_lg.ppp)~1,
                                  clusters = "LGCP", model = "exponential")
coco_wf_hum_sm_naive.kppm <- kppm(unmark(coco_wf_hum_sm.ppp)~1,
                                  clusters = "LGCP", model = "exponential")
coco_wf_nat_naive.kppm <- kppm(unmark(coco_wf_nat.ppp)~1,
                               clusters = "LGCP", model = "exponential")
coco_wf_nat_lg_naive.kppm <- kppm(unmark(coco_wf_nat_lg.ppp)~1,
                                  clusters = "LGCP", model = "exponential")
coco_wf_nat_sm_naive.kppm <- kppm(unmark(coco_wf_nat_sm.ppp)~1,
                                  clusters = "LGCP", model = "exponential")

plot_envelopes <- function(naive.kppm, model.kppm, title){
  plot(envelope(naive.kppm, fun = Fest, verbose=F),
     col = c("black","darkgreen", NA, NA),
     shadecol = adjustcolor("gray", 0.5),
     lwd = 2, legend = F, main = title)
  plot(envelope(model.kppm, fun = Fest, verbose=F),
     col = c(NA, "darkred", NA, NA),
     shadecol = adjustcolor("gray", 0.5),
```

```
    lwd = 2, add = T)
  legend("topleft", legend=c("Intercept", "KPPM"),
       col=c("darkgreen", "darkred"), lty=2)

  plot(envelope(naive.kppm, fun = Gest, verbose=F),
     col = c("black","darkgreen", NA, NA),
     shadecol = adjustcolor("gray", 0.5),
     lwd = 2, legend = F, main = title)
  plot(envelope(model.kppm, fun = Fest, verbose=F),
     col = c(NA, "darkred", NA, NA),
     shadecol = adjustcolor("gray", 0.5),
     lwd = 2, add = T)
  legend("topleft", legend=c("Intercept", "KPPM"),
       col=c("darkgreen", "darkred"), lty=2)
}
```

Human Caused

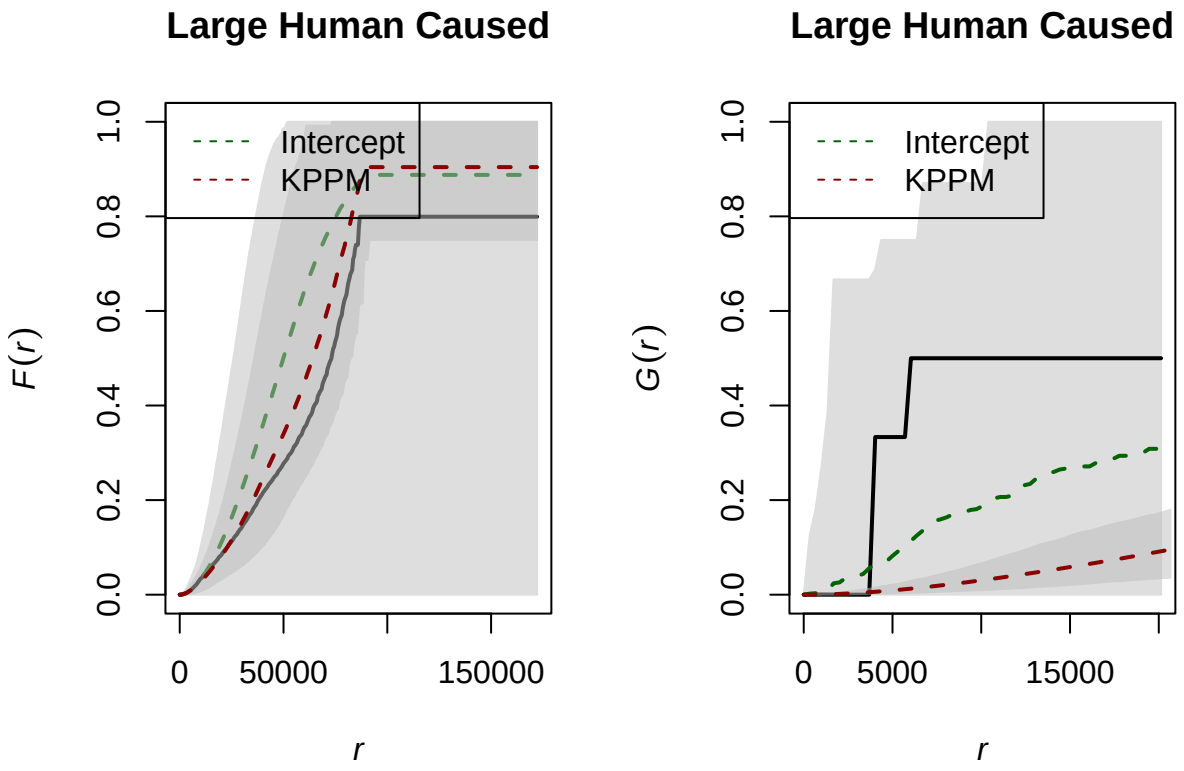
All

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_naive.kppm, coco_wf_hum.kppm, "All Human Caused")
```

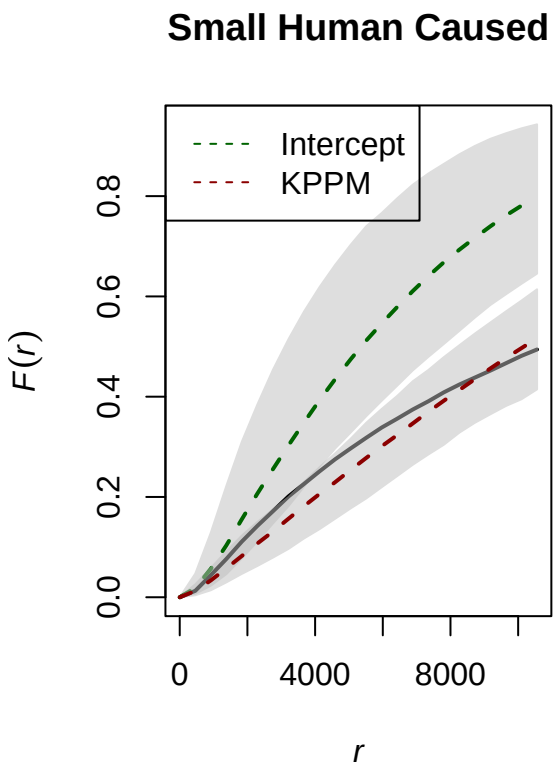


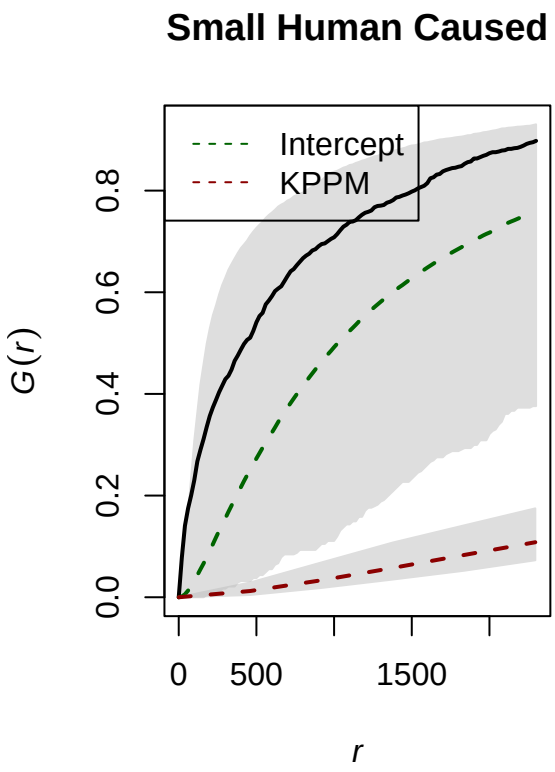
Threshold

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_lg_naive.kppm, coco_wf_hum_lg.kppm, "Large Human Caused")
```




```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_sm_naive.kppm, coco_wf_hum_sm.kppm, "Small Human Caused")
```

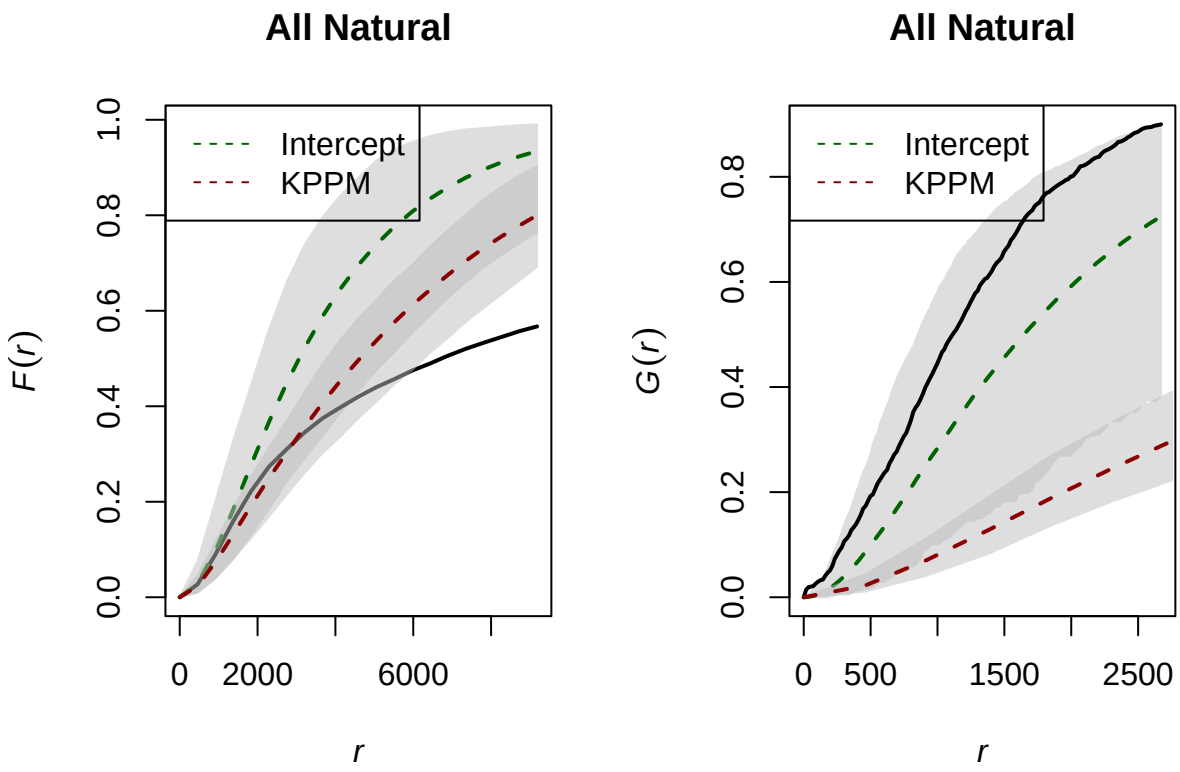




Natural

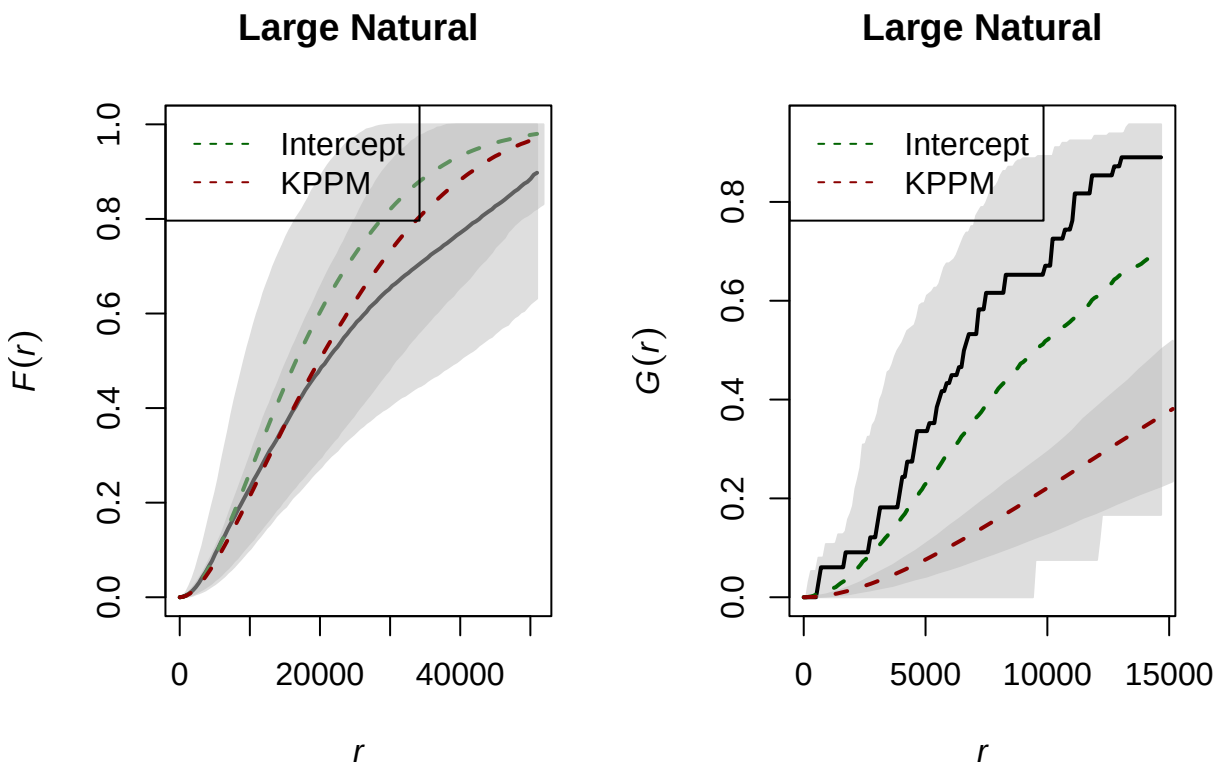
All

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_naive.kppm, coco_wf_nat.kppm, "All Natural")
```

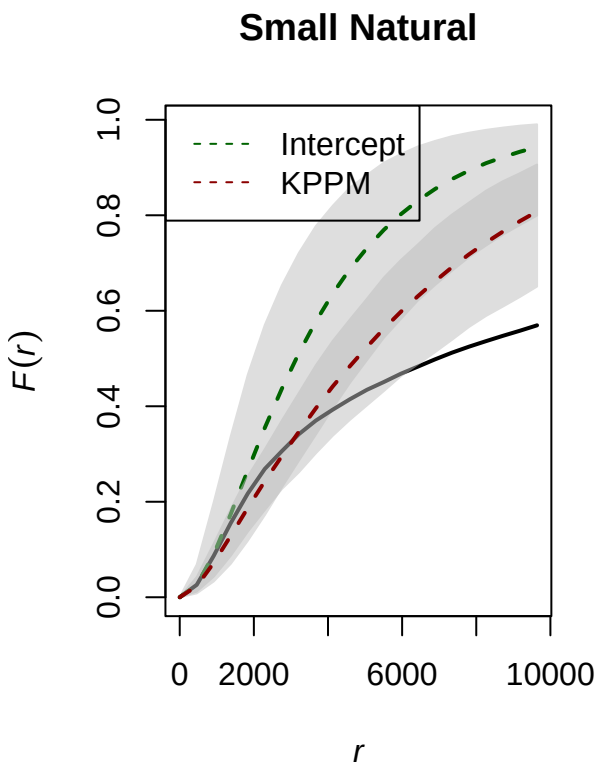


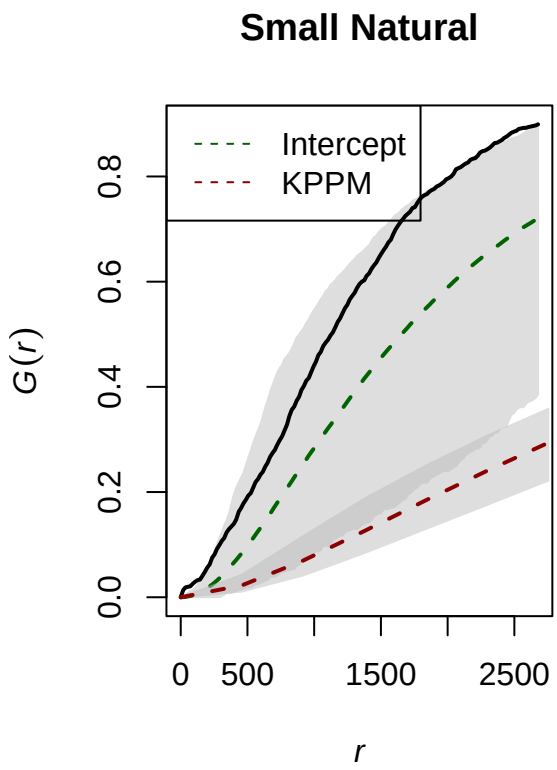
Threshold

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_lg_naive.kppm, coco_wf_nat_lg.kppm, "Large Natural")
```




```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_sm_naive.kppm, coco_wf_nat_sm.kppm, "Small Natural")
```





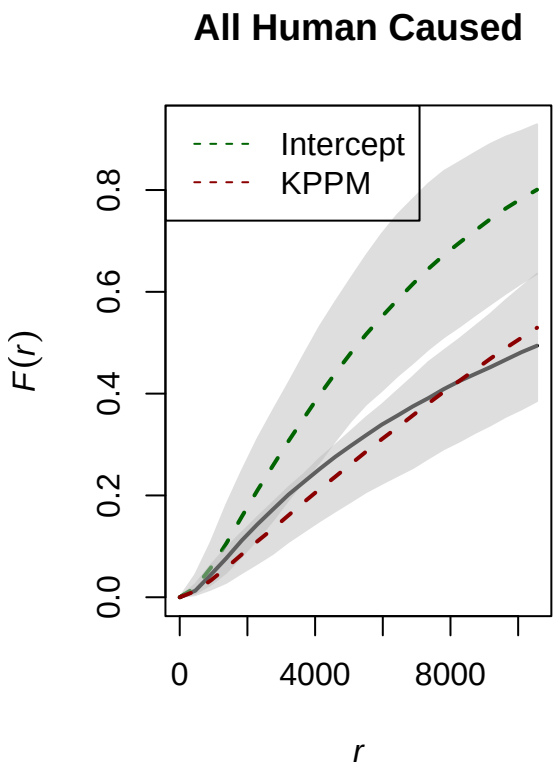
All Parameters

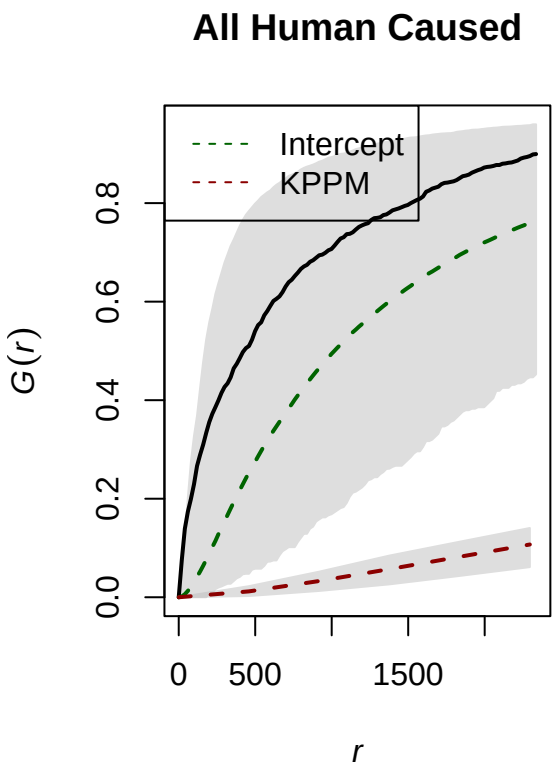
```
rbind("All Human Caused" = coco_wf_hum.kppm$par,
      "Large Human Caused" = coco_wf_hum_lg.kppm$par,
      "Small Human Caused" = coco_wf_hum_sm.kppm$par,
      "All Natural" = coco_wf_nat.kppm$par,
      "Large Natural" = coco_wf_nat_lg.kppm$par,
      "Small Natural" = coco_wf_nat_sm.kppm$par)
```

```
##                          sigma2     alpha
## All Human Caused   4.728370e+00  7791.704
## Large Human Caused 3.220770e-09 29691.296
## Small Human Caused 4.688047e+00  7806.519
## All Natural        2.008119e+00 12000.200
## Large Natural      1.400144e+00  5681.422
## Small Natural      2.056924e+00 12054.552
```

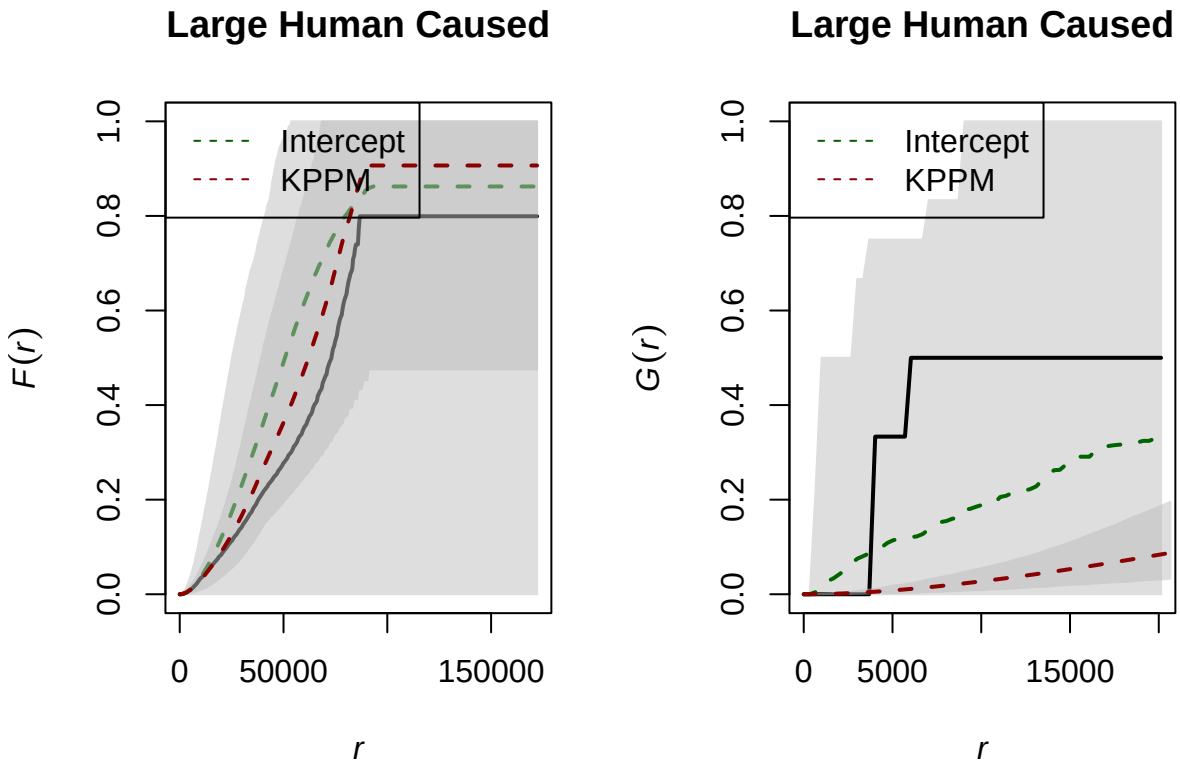
All Plots

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_naive.kppm, coco_wf_hum.kppm, "All Human Caused")
```

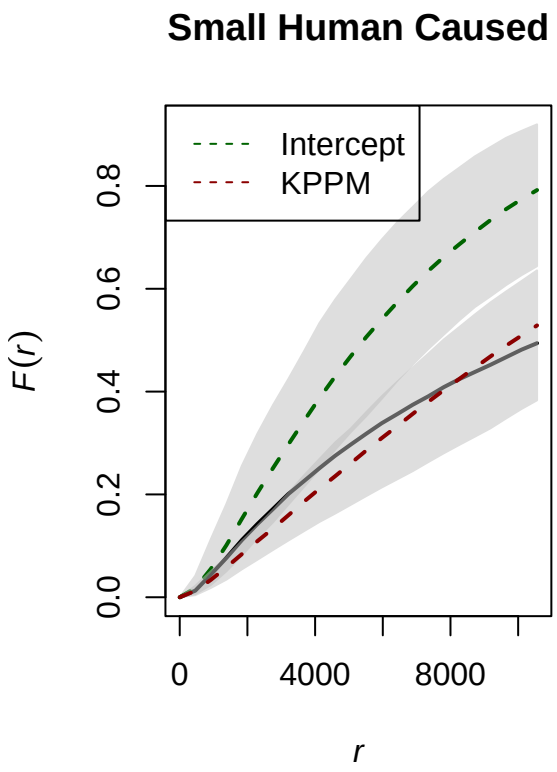


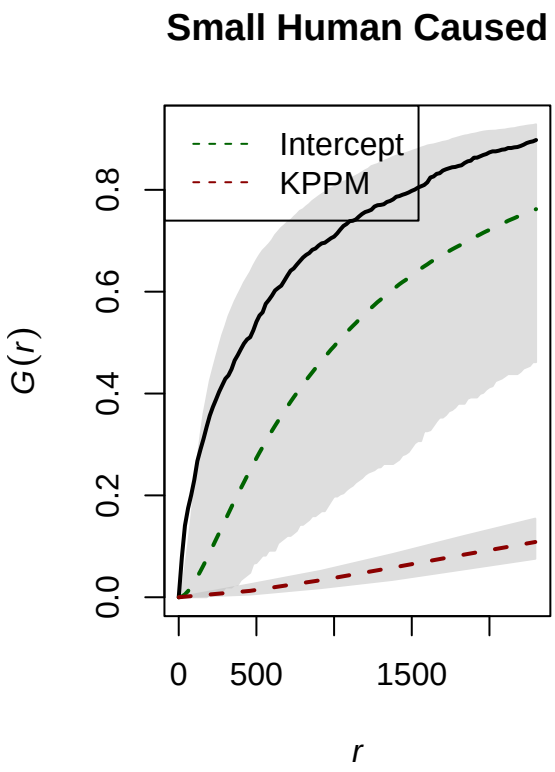



```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_lg_naive.kppm, coco_wf_hum_lg.kppm, "Large Human Caused")
```

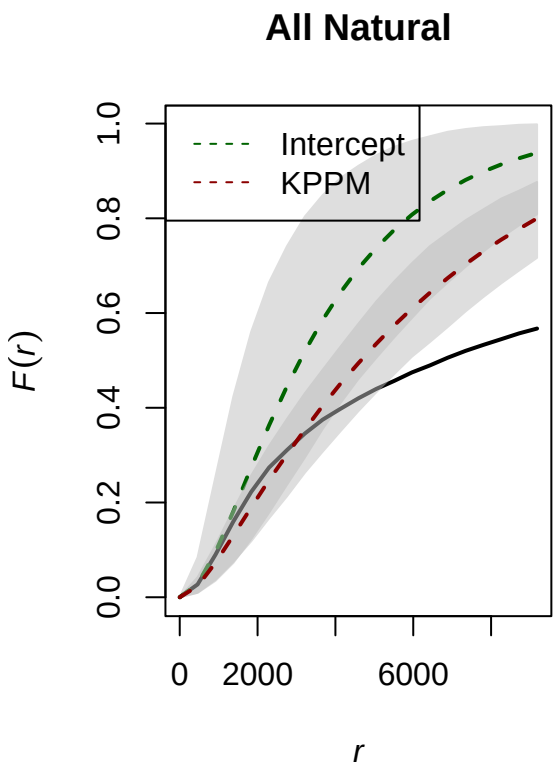


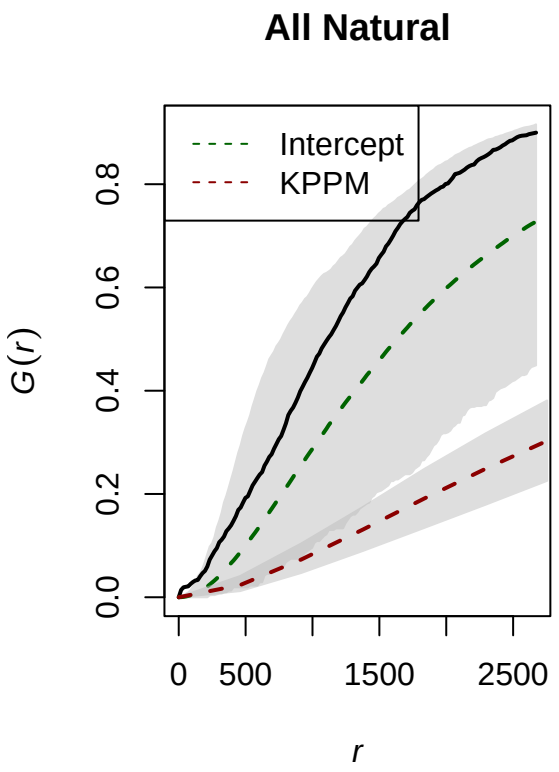

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_hum_sm_naive.kppm, coco_wf_hum_sm.kppm, "Small Human Caused")
```



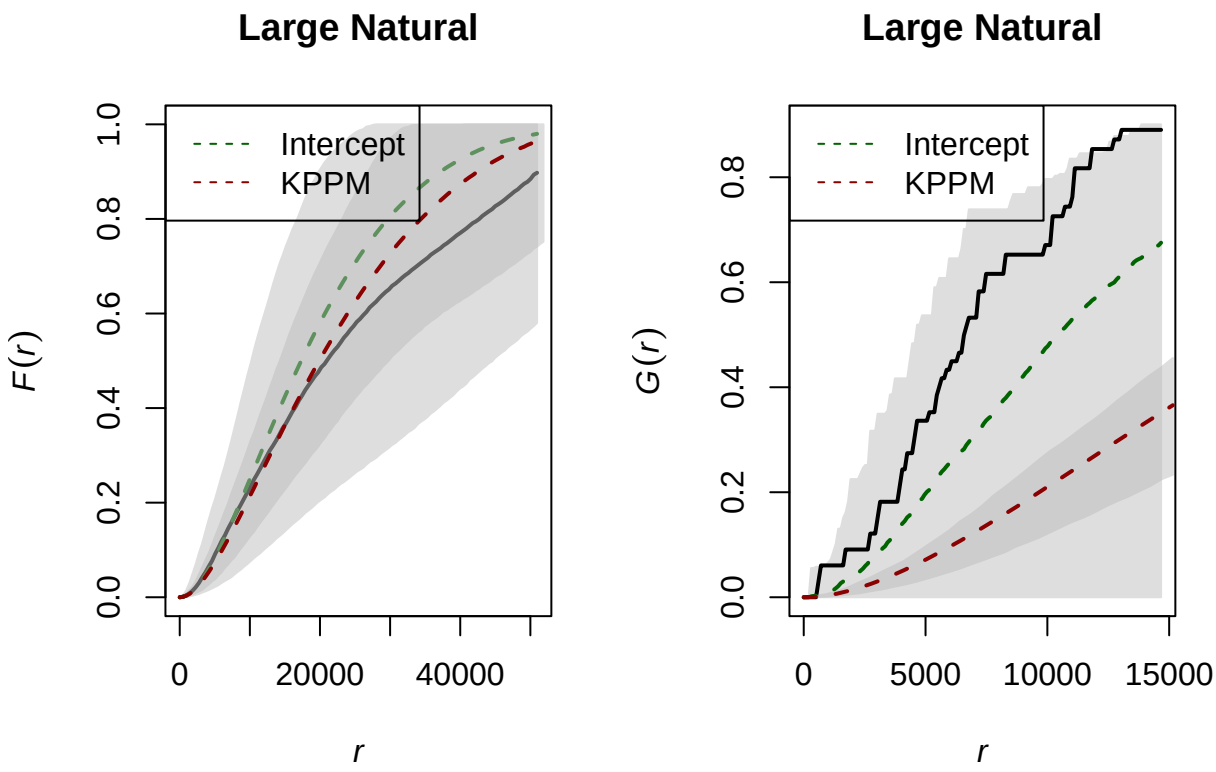



```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_naive.kppm, coco_wf_nat.kppm, "All Natural")
```

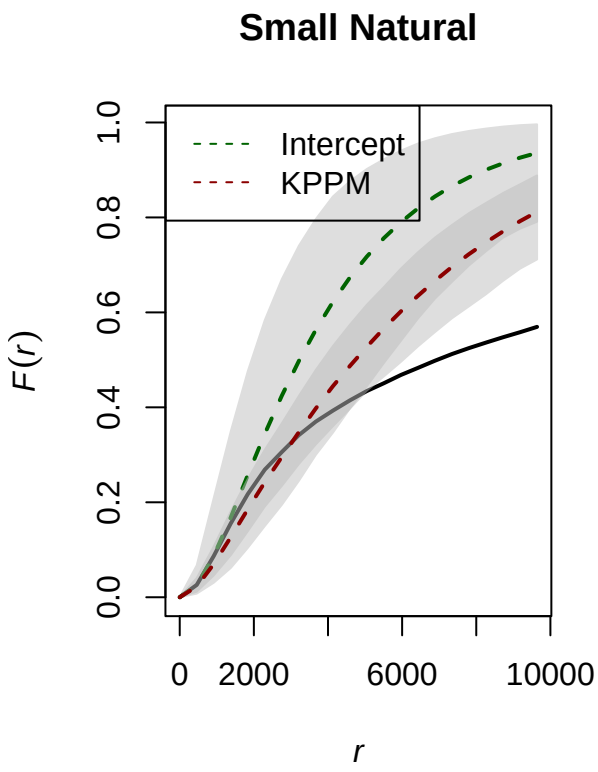


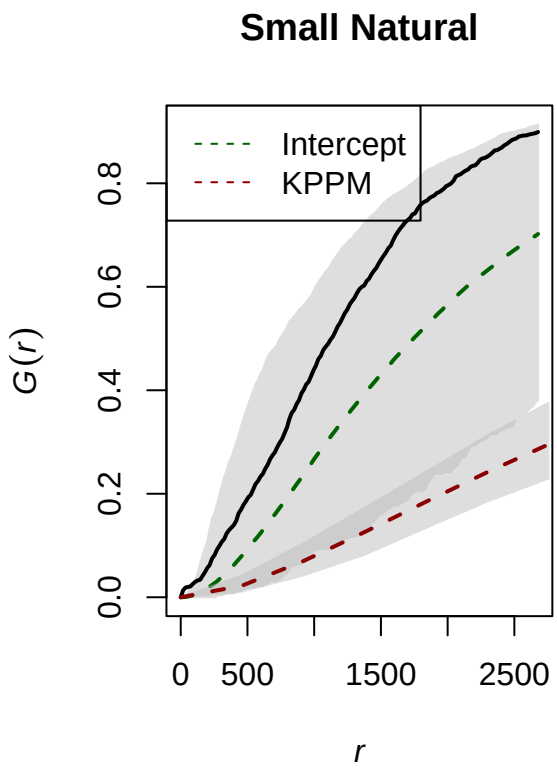



```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_lg_naive.kppm, coco_wf_nat_lg.kppm, "Large Natural")
```



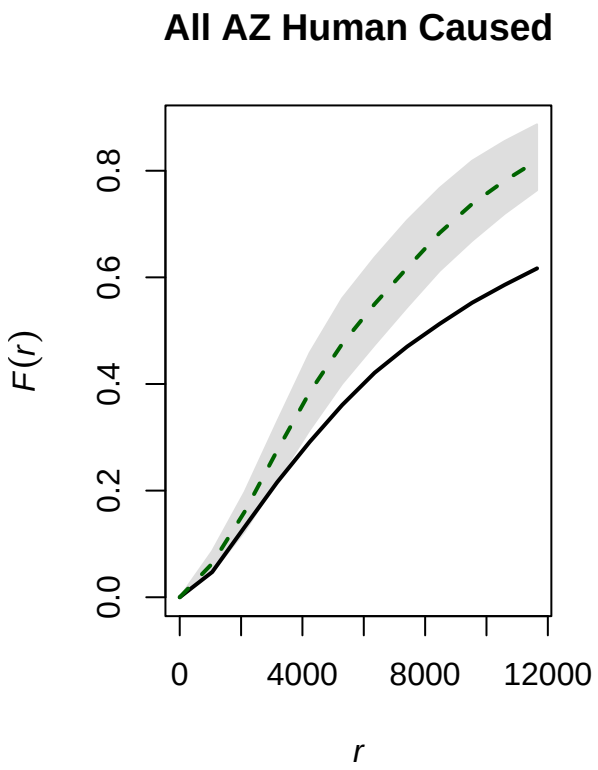

```
par(mfrow = c(1, 2))
plot_envelopes(coco_wf_nat_sm_naive.kppm, coco_wf_nat_sm.kppm, "Small Natural")
```

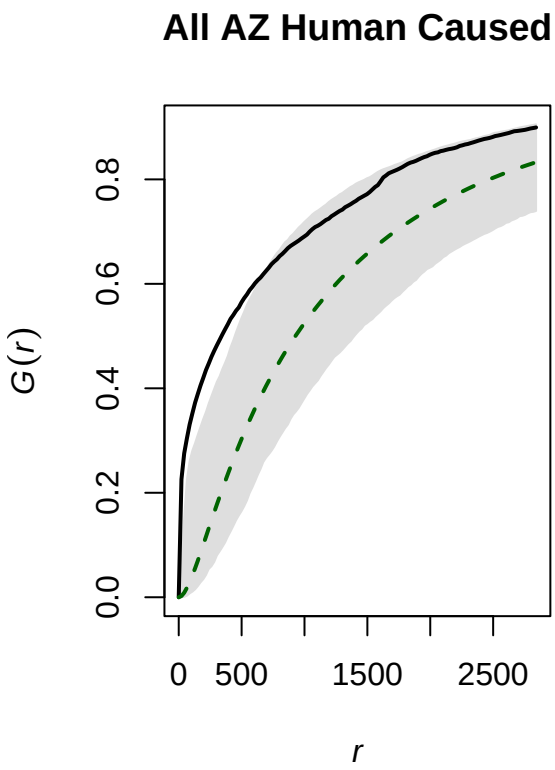




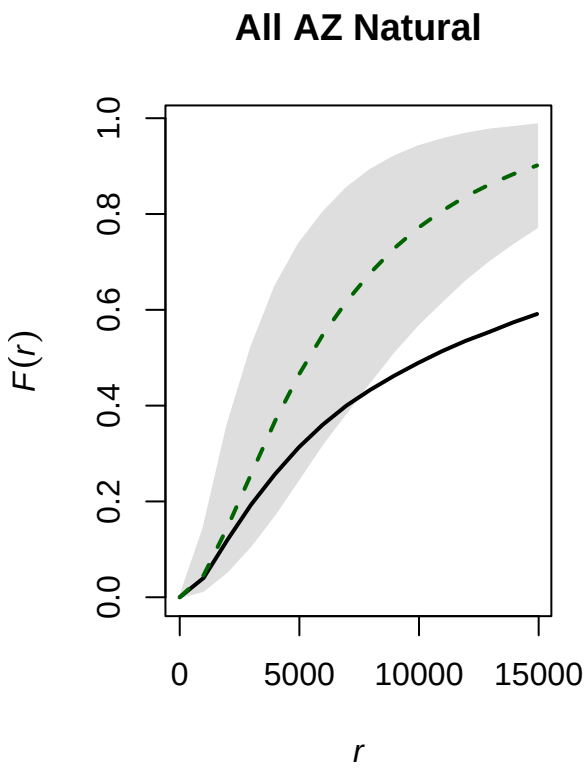
All of Arizona CSR

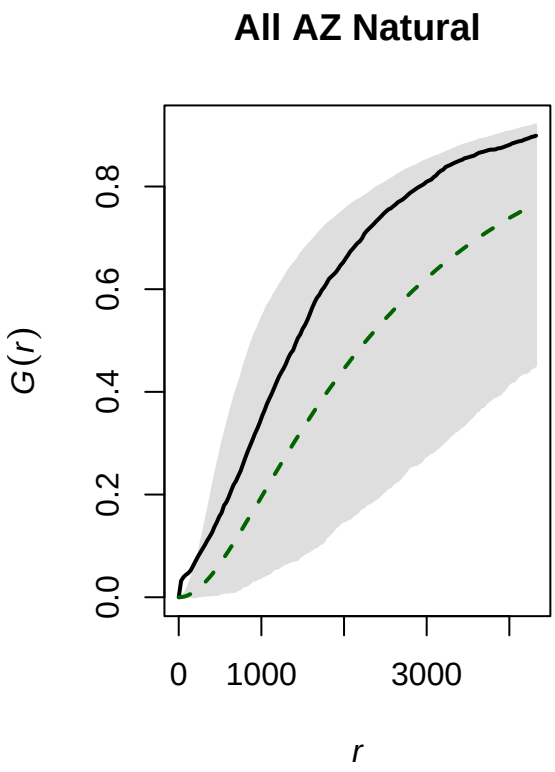
```
az_wf_hum.ppp <- as.ppp(az_fires_wf_hum)
az_wf_nat.ppp <- as.ppp(az_fires_wf_nat)

az_wf_hum_naive.kppm <- kppm(unmark(az_wf_hum.ppp)~1,
                             clusters = "LGCP", model = "exponential")
az_wf_nat_naive.kppm <- kppm(unmark(az_wf_nat.ppp)~1,
                             clusters = "LGCP", model = "exponential")

par(mfcol=c(1,2))
plot(envelope(az_wf_hum_naive.kppm, fun = Fest, verbose=F),
   col = c("black","darkgreen", NA, NA),
   shadecol = adjustcolor("gray", 0.5),
   lwd = 2, legend = F, main = "All AZ Human Caused")
plot(envelope(az_wf_hum_naive.kppm, fun = Gest, verbose=F),
   col = c("black","darkgreen", NA, NA),
   shadecol = adjustcolor("gray", 0.5),
   lwd = 2, legend = F, main = "All AZ Human Caused")
```






```
par(mfcol=c(1,2))
plot(envelope(az_wf_nat_naive.kppm, fun = Fest, verbose=F),
   col = c("black","darkgreen", NA, NA),
   shadecol = adjustcolor("gray", 0.5),
   lwd = 2, legend = F, main = "All AZ Natural")
plot(envelope(az_wf_nat_naive.kppm, fun = Gest, verbose=F),
   col = c("black","darkgreen", NA, NA),
   shadecol = adjustcolor("gray", 0.5),
   lwd = 2, legend = F, main = "All AZ Natural")
```


All AZ Natural
F(r)
0.0
0.2
0.4
0.6
0.8
1.0
0
5000
10000
15000
r


All AZ Natural
G(r)
0.0
0.2
0.4
0.6
0.8
0
1000
3000
r

Appendix C

Alex Salce

```
library(dplyr)
library(tigris)
library(sf)
library(ggplot2)
library(mapview)
library(tidyverse)
library(scales)
library(knitr)
library(readr)
```

Packages we need.

We have a few datasets that we have pre-constructed based on available resources and challenges with acquiring data for the features that we are interested in modeling. Note that all `sf` objects are projected to `EPSG:3612` (https://epsg.io/32612).

- `wfigs_az_sf` is a `sf` object with 18089 observations of wildfire incidence in the state of Arizona. This data was originally acquired via Wildland Fire Incident Locations from the National Interagency Fire Center. It contains spatial point data indicating the origin of each wildfire recorded in the IRWIN database, and includes many useful features. Most notably, the data includes the `IncidentSize`, which is the size in acres of the resulting wildfire. `IncidentSize` is the primary response variable of interest for our study. The data included in this dataframe includes hand-selected features that were deemed to be potentially useful during the exploration phase of the project, as well as other data manually recovered for natural and environmental factors, distances to roads in Arizona, as well as census data for population density, discussed elsewhere in the report.

-`az_rd_primary` and `az_rd_secondary` are `sf` data of all roads in Arizona with FCC road classification codes of `S1100` or `S1200`. These two classes are treated as "major" roads in our analyses. `az_rd_4WD` is all roads with `S1500` classification, and are representative of remote roadways in Arizona. All other road types are removed/disregarded as they are deemed not useful for our analysis. See this document for all classification codes.

```
arizona_sf <- states() %>% filter_state("arizona")
arizona_sf <- st_transform(arizona_sf, crs = "EPSG:32612")

az_counties_sf <- counties(state = "AZ", cb = TRUE)
az_counties_sf <- st_transform(az_counties_sf, crs = "EPSG:32612")

# make national forest objects

coconino_nf_sf <- naz_forests_sf[naz_forests_sf$FORESTNAME=="Coconino National Forest",]

kaibab_nf_sf <- naz_forests_sf[naz_forests_sf$FORESTNAME=="Kaibab National Forest",]
```

First, we are going to load the `sf` object for the state of Arizona and project to `EPSG:32612`.

Create Wildfire Dataframes

```
az_fires_rx <- wfigs_az_sf %>% filter(IncidentTypeCategory=="RX")
az_fires_wf <- wfigs_az_sf %>% filter(IncidentTypeCategory=="WF")

az_fires_wf_hum <- az_fires_wf %>% filter(FireCause=="Human")
az_fires_wf_nat <- az_fires_wf %>% filter(FireCause=="Natural")
az_fires_wf_un <- az_fires_wf %>% filter(FireCause=="Undetermined" | FireCause=="Unknown")
```

For our analyses using `wfigs_az_sf`, we want to be sure to differentiate between fires that are prescribed burns (`IncidentTypeCategory=="RX"`) and wildfires (IncidentTypeCategory==“WF”). Our analyses will only use fires of type `WF`, since prescribed burns are fires deliberately started and controlled by a service entity.

We also want to differentiate fires by their causes. There are four unique categories of fire in the `FireCause` column, `HUMAN`, `NATURAL`, `UNDETERMINED`, and `UNKNOWN`. For our analyses, we will discard `UNDETERMINED` and `UNKNOWN` type fires, as we cannot reasonably assume anything about them. Additionally, they comprise only about 12% of all of the `wfigs_az_sf` data.

Histograms

Nearest Roads

This histogram uses input `sf` object to create `histdf`, which is used to generate a histogram of the number of closest roads, either primary/secondary roads or 4wd roads, to each wildfire incident origin. The x-axis indicates the distance in meters of that bin to wildfire points.

As a generalization, we will think of Primary & Secondary roads as essentially the same class of roads, and we know that many humans use these roads every day. No matter the stretch of road, there is a good chance of there being some human settlement of some kind alongside these roads within close proximity.

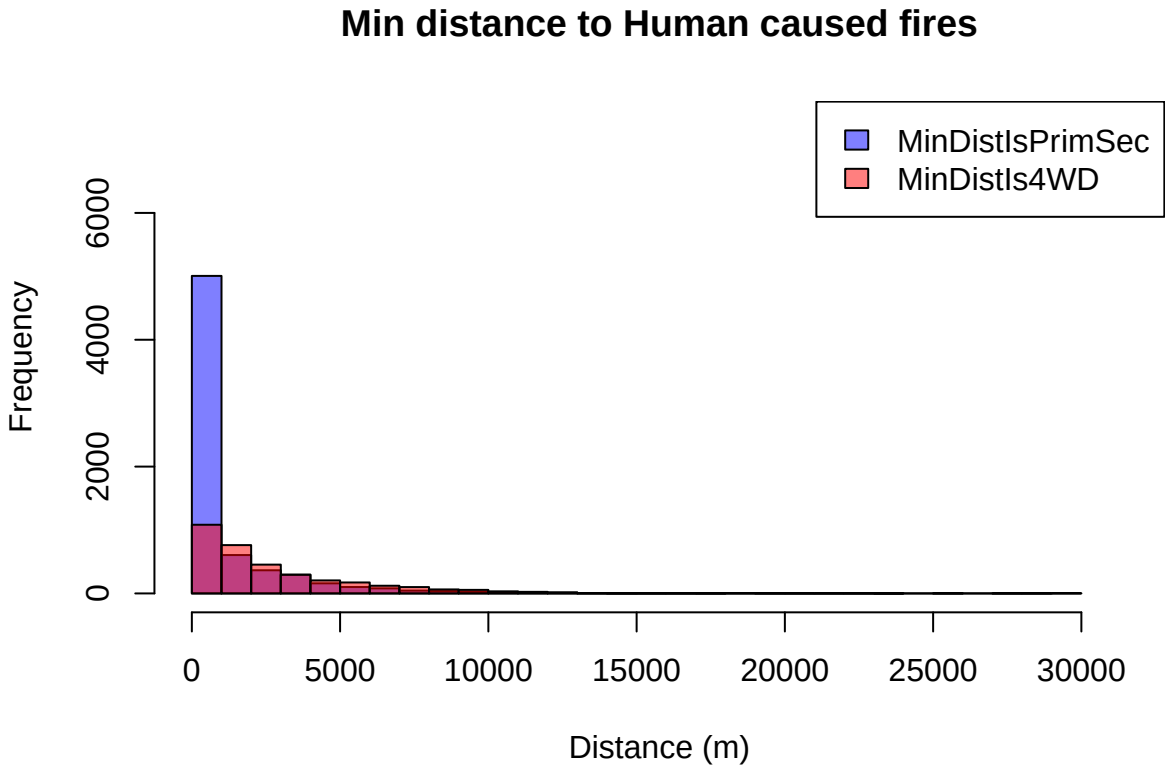
The “4WD” roads are technically a fairly “incomplete” dataset, but they do give a sense of areas humans can spend their time in remote areas. In general, though, if a wildfire incident is closer to a 4WD road we can think of it as being a “more remote” fire, i.e. there is a good chance it’s fairly removed from any prominent human settlement.

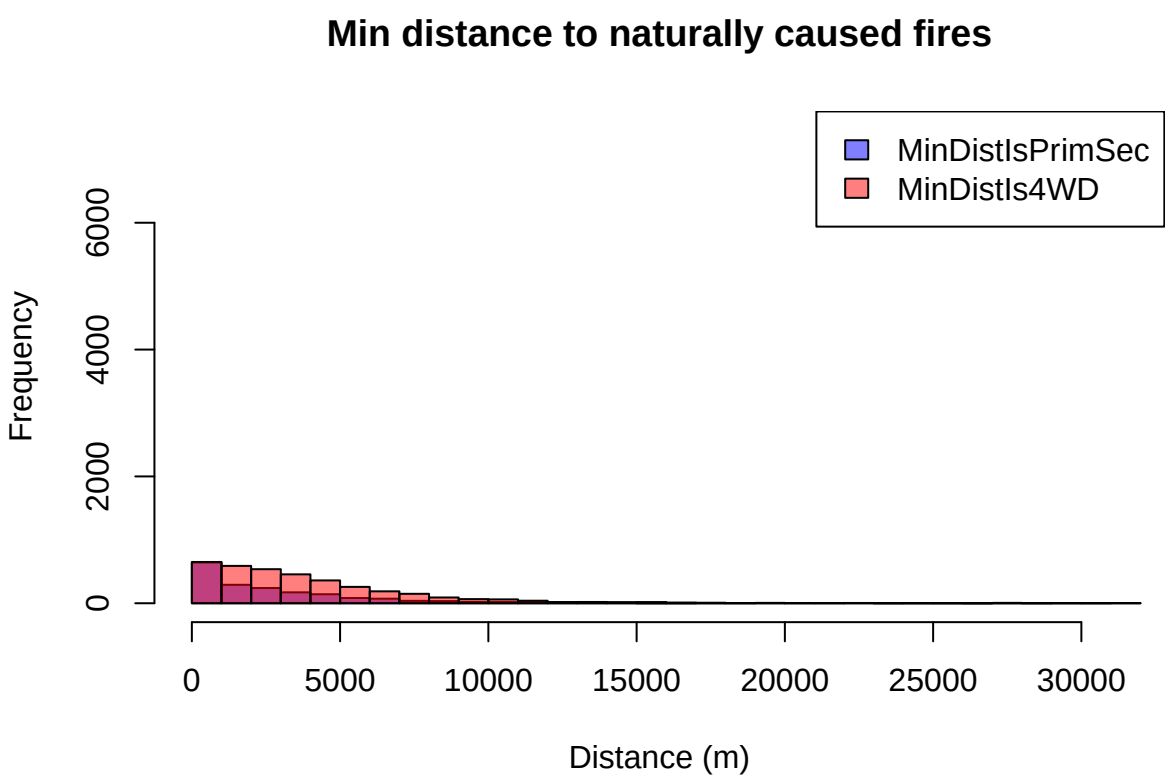
For reference, the minimum distances as well as the closest road type indicators were generated using the below code.

```
wfigs_az_sf$distance_rd_primary <-
  st_distance(wfigs_az_sf, az_rd_primary) %>% apply(1, min)

wfigs_az_sf$distance_rd_secondary <-
  st_distance(wfigs_az_sf, az_rd_secondary) %>% apply(1, min)

wfigs_az_sf$distance_rd_4wd <-
  st_distance(wfigs_az_sf, az_rd_4wd) %>% apply(1, min)


wfigs_az_sf <- wfigs_az_sf %>%
  mutate(distance_rd_min_prisec = pmin(distance_rd_primary,
                                       distance_rd_secondary))

wfigs_az_sf <- wfigs_az_sf %>%
  mutate(distance_rd_min_all = pmin(distance_rd_primary,
                                    distance_rd_secondary,
                                    distance_rd_4wd))
```

```
wfigs_az_sf$distance_rd_min_isprisec <- as.integer(wfigs_az_sf$distance_rd_min_all ==
                                                   wfigs_az_sf$distance_rd_min_prisec)
```

All wildfire data We will first look at the "Nearest Roads" histograms for all wildfire data (no filtering for the size of the fire.)





WildFireType	Counts
Human Caused Fires	10174
Naturally Caused Fires	5409
Unknown Caused Fires	2210

There is a very high concentration of wildfires cause by humans very close to primary and secondary roads, and natural fires generally seem to be more remote (in general closer to more 4WD roads.)

```
# Extract Coconino County
coconino_sf <- az_counties_sf %>%
  filter(NAME == "Coconino")

az_fires_wf_hum <- az_fires_wf %>% filter(FireCause=="Human")
az_fires_wf_nat <- az_fires_wf %>% filter(FireCause=="Natural")
az_fires_wf_un <- az_fires_wf %>% filter(FireCause=="Undetermined" | FireCause=="Unknown")

az_fires_wf_hum <- st_intersection(az_fires_wf_hum, coconino_sf)
```

Coconino County Wildfire Data

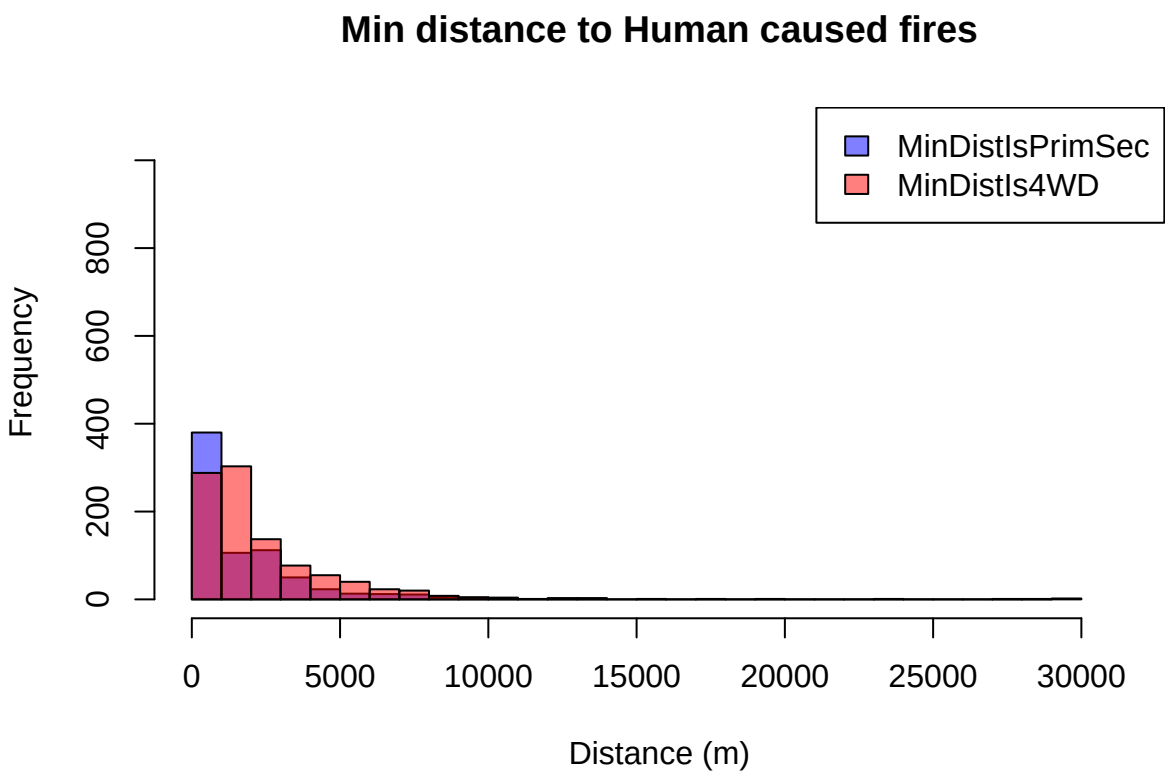
```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

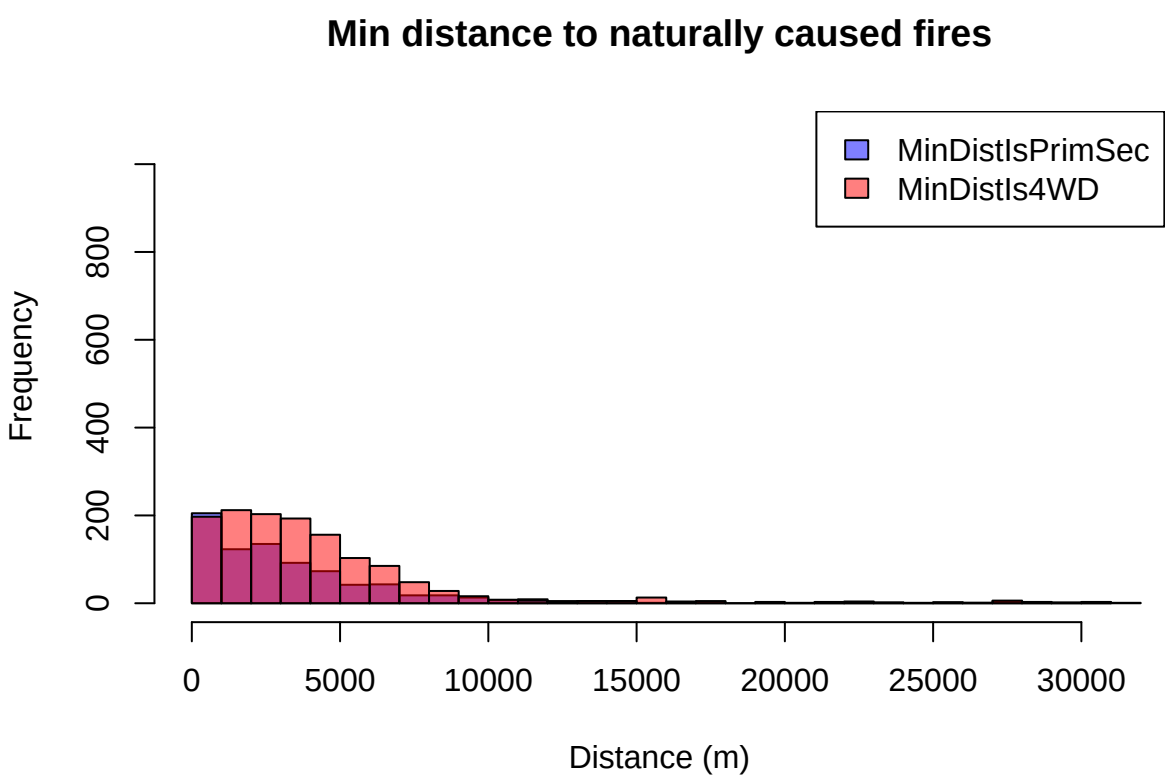
```r
az_fires_wf_nat <- st_intersection(az_fires_wf_nat, coconino_sf)
```

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

```r
az_fires_wf_un <- st_intersection(az_fires_wf_un, coconino_sf)
```

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```





WildFireType	Counts
Human Caused Fires	1692
Naturally Caused Fires	2128
Unknown Caused Fires	296

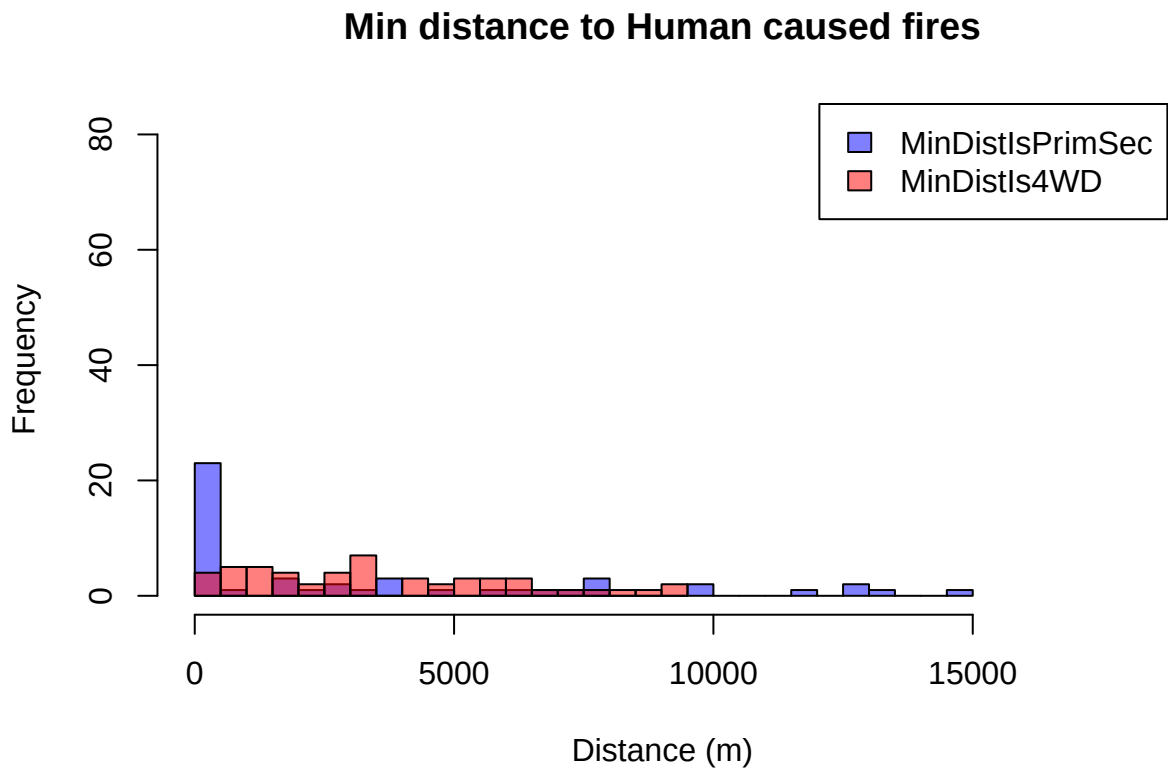
Thresholded for Wildfire Acreage We define the threshold for a fire to be "large" as `IncidentSize` ≥ 1000 acres. We threshold the data accordingly and replot our histograms.

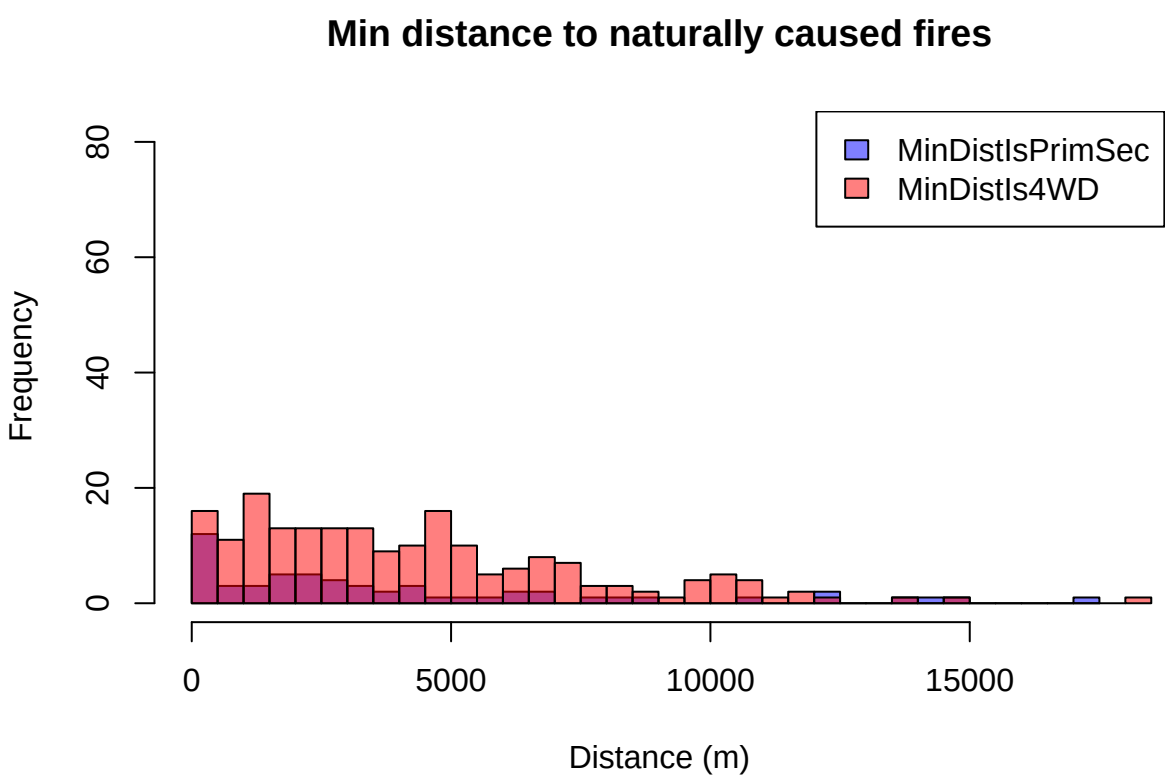
```
####################################
# RUN BELOW FOR FIRE SIZE THRESHOLD#
####################################


# fire size threshold
# FIRE_SIZE_CLASS = Code for fire size based on the number of acres within the
# final fire perimeter (A=greater than 0 but less than or equal to 0.25 acres,
# B=0.26-9.9 acres, C=10.0-99.9 acres, D=100-299 acres, E=300 to 999 acres,
# F=1000 to 4999 acres, and G=5000+ acres).

# class F and G fires
wf_size_threshold <- 1000


wfigs_az_sf$size_threshold <- as.integer(wfigs_az_sf$IncidentSize >= wf_size_threshold)
```

```r
wfigs_az_sf_thresh <- wfigs_az_sf %>% filter(size_threshold==1)


az_fires_rx <- wfigs_az_sf_thresh %>% filter(IncidentTypeCategory=="RX")
az_fires_wf <- wfigs_az_sf_thresh %>% filter(IncidentTypeCategory=="WF")


#human caused wildfires

table(wfigs_az_sf_thresh$FireCause)
```

```
##
##        Human      Natural Undetermined      Unknown
##          102          255           62           20
```

```r
az_fires_wf_hum <- az_fires_wf %>% filter(FireCause=="Human")
az_fires_wf_nat <- az_fires_wf %>% filter(FireCause=="Natural")
az_fires_wf_un <- az_fires_wf %>% filter(FireCause=="Undetermined" | FireCause=="Unknown")
```





WildFireType	Counts
Human Caused Fires	101
Naturally Caused Fires	255
Unknown Caused Fires	33

Models

Spatial Linear Model

$$\mathbf{y} = \mathbf{X}^T\boldsymbol{\beta} + \boldsymbol{e} \quad \boldsymbol{e} \sim \mathrm{N}(\mathbf{0}, \Sigma(\boldsymbol{\theta}))$$

Log-Gaussian Cox Proces

$$\log(\lambda(u)) = \boldsymbol{Z}(u)\boldsymbol{\beta} + e(u), \quad e(u) \sim \mathrm{N}(0, C(\boldsymbol{\theta})), \quad C(u, u') = \sigma^2 e^{-||u-u'||/h}$$

Point Process Binary Spatial GLM Logistic Regression

$$\mathrm{logit}(\lambda_1(\boldsymbol{s})) = \mathbf{x}(\boldsymbol{s})^T\boldsymbol{\beta} + e(\boldsymbol{s}) + \log(\lambda_0), \quad Y(\boldsymbol{s}) \sim \mathrm{Bern}(p(\boldsymbol{s})), \quad \mathrm{E}[Y(\boldsymbol{s})] = p(\boldsymbol{s}) = \frac{\lambda_1(\boldsymbol{s})}{\lambda_0(\boldsymbol{s}) + \lambda_1(\boldsymbol{s})}$$

```
####################################
# RUN BELOW FOR FIRE SIZE THRESHOLD#
####################################
```

```
# fire size threshold
# FIRE_SIZE_CLASS = Code for fire size based on the number of acres within the
# final fire perimeter (A=greater than 0 but less than or equal to 0.25 acres,
# B=0.26-9.9 acres, C=10.0-99.9 acres, D=100-299 acres, E=300 to 999 acres,
# F=1000 to 4999 acres, and G=5000+ acres).

# class F and G fires
wf_size_threshold <- 1000


wfigs_az_sf$size_threshold <- as.integer(wfigs_az_sf$IncidentSize >= wf_size_threshold)
# wfigs_az_sf$size_threshold <- as.integer(wfigs_az_sf$IncidentSize < wf_size_threshold)

wfigs_az_sf_thresh <- wfigs_az_sf %>% filter(size_threshold==1)


az_fires_rx <- wfigs_az_sf_thresh %>% filter(IncidentTypeCategory=="RX")
az_fires_wf <- wfigs_az_sf_thresh %>% filter(IncidentTypeCategory=="WF")


#human caused wildfires

table(wfigs_az_sf_thresh$FireCause)
```

Threshold for large wildfires

```
##
##        Human      Natural Undetermined      Unknown
##          102          255           62           20
```

```
az_fires_wf_hum <- az_fires_wf %>% filter(FireCause=="Human")
az_fires_wf_nat <- az_fires_wf %>% filter(FireCause=="Natural")
az_fires_wf_un <- az_fires_wf %>% filter(FireCause=="Undetermined" | FireCause=="Unknown")
```

The first step is to filter our data to only "large" wildfires (IncidentSize ≥ 1000 acres). We are now treating our data as point process data, so the magnitude of the resulting wildfire is no longer of interest. Rather, we will be only studying the occurrence of "big" wildfires.

Since we will be treating the wildfires as a point process, the most sensible data to study should be the naturally occurring fires as it would be reasonable to assume that they occur randomly. For example, naturally occurring fires can be the result of lightning strikes, which we have previously seen modeled using a point process approach.

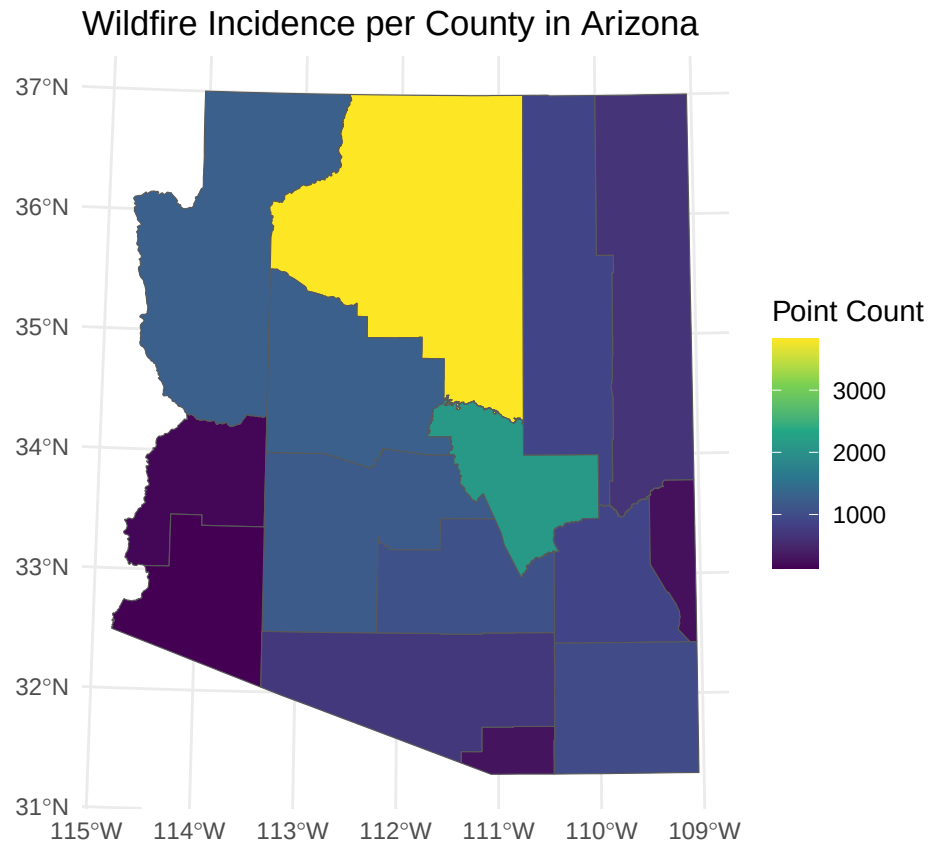
```
# az_county_intersections <- st_intersection(az_counties_sf, rbind(az_fires_wf_hum, az_fires_wf_nat))

# az_county_intersections <- st_intersection(az_counties_sf, az_fires_wf_nat)
az_county_intersections <- st_intersection(az_counties_sf, wfigs_az_sf[wfigs_az_sf$FireCause=="Human"|wf
```

Counties

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

```
county_counts <- az_county_intersections %>%
  group_by(NAME) %>%
  summarise(count = n())
```

```r
az_counties_with_counts <- az_counties_sf %>%
  st_join(county_counts, by = "NAME") %>%
  mutate(count = replace_na(count, 0))


ggplot(az_counties_with_counts) +
  geom_sf(aes(fill = count)) +
  scale_fill_viridis_c(name = "Point Count") +
  theme_minimal() +
  labs(title = "Wildfire Incidence per County in Arizona")
```




```r
rm(county_counts, az_counties_with_counts, az_county_intersections)
```

```r
#filter large wildfire data to only Coconino County
az_fires_wf <- st_intersection(az_fires_wf[az_fires_wf$FireCause=="Human" | az_fires_wf$FireCause == "Na
```

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

```r
#filter naz national forest sf objects
naz_forests_sf <- st_intersection(naz_forests_sf, coconino_sf)
```

```
## Warning: attribute variables are assumed to be spatially constant throughout
## all geometries
```

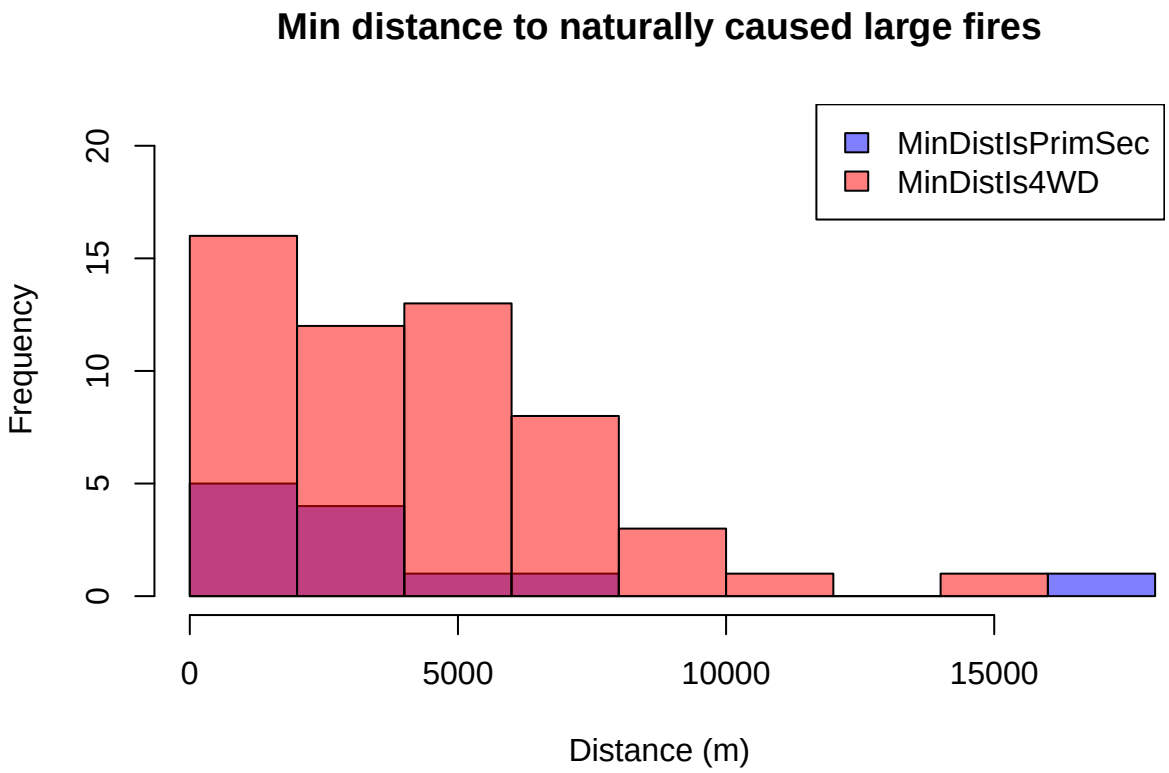
For this portion of the study, we opted to start smaller by filtering the data down to only Coconino County. Coconino County has the most naturally occurring wildfires in the state by a good margin, and is home to part of the largest contiguous Ponderosa Pine forest in the United States.

Model

```
library(spatstat)
library(spmodel)
library(lubridate)
library(tidycensus)
library(ggmap)
```

Additional Packages

Data Refinement & Additional Exploration Minimum distance to naturally caused fires in Coconino County.



WildFireType	Counts
Human Caused Fires	6
Naturally Caused Fires	66
Unknown Caused Fires	0

In general, the naturally occurring fires that end up becoming large are rather remote, as indicated by more roads that are in closer proximity to 4WD roads, and even among those, many of which are over a kilometer away from even the nearest remote roads.

Building a Background Poisson Process Realization We will assign `az_fires_wf_spglm` as a placeholder `sf` object for whichever data we are interested in (for flexibility). For the report, we are only analyzing naturally caused "large" wildfires, hence it will be assigned using `az_fires_wf_nat` to generate ppp object and subsequent

Poisson background realization points. Note that we are following a similar process as outlined in the *Random spatial index (point process) + Binary GLM* application in the *Non-Gaussian spatial data* lecture for the `gorillas` data.

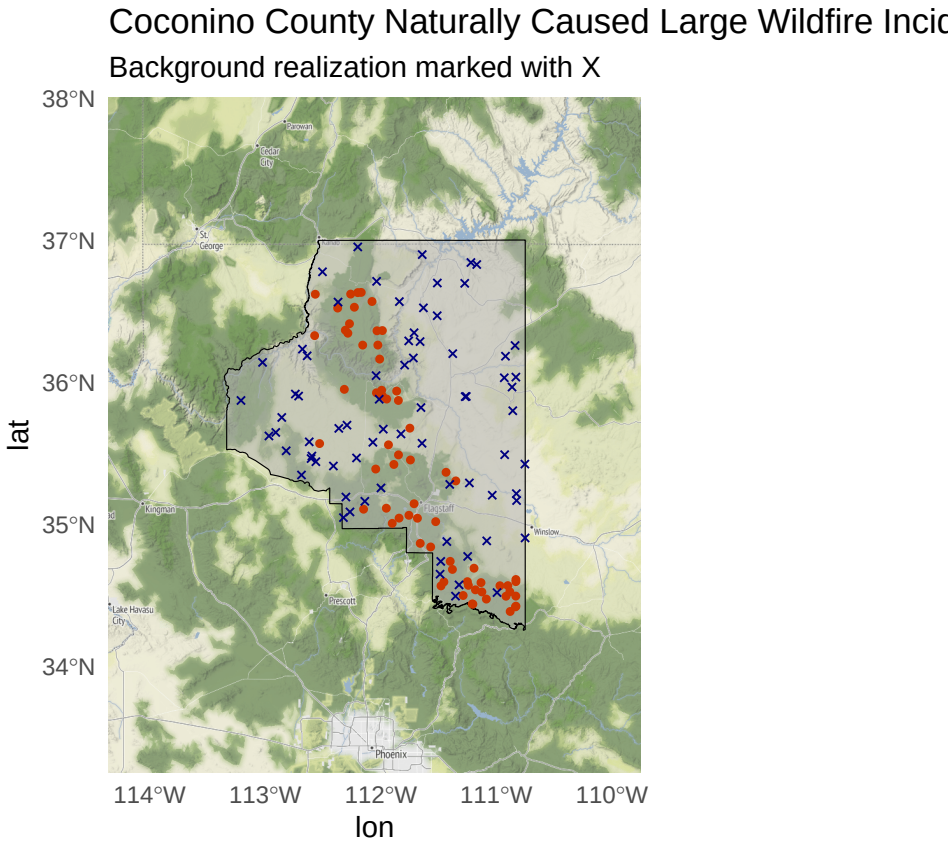
```
# set seed to 219 for report!
set.seed(219)

# set seed to 20 for predictions!
# set.seed(20)

# pick df of interest

az_fires_wf_spglm <- az_fires_wf_nat


az_fires_wf_spglm_ppp <- as.ppp(az_fires_wf_spglm)

background <- rpoispp((az_fires_wf_spglm_ppp$n) / area(as.owin(az_fires_wf_spglm_ppp)),
                      win = as.owin(coconino_sf))

df <- data.frame(x = background$x, y = background$y)

background_sf <- st_as_sf(df, coords = c("x", "y"), crs = "EPSG:32612")
```

```
wf_sf_cc_plot <- st_as_sf(as.data.frame(az_fires_wf_spglm_ppp), coords = c("x", "y"), crs="EPSG:32612")

wf_sf_cc_plot <- st_transform(wf_sf_cc_plot, crs = 4326)

bkg_sf_cc_plot <- st_as_sf(as.data.frame(background), coords = c("x", "y"), crs="EPSG:32612")

bkg_sf_cc_plot <- st_transform(bkg_sf_cc_plot, crs = 4326)

coconino_plot <- coconino_sf %>% st_transform(4326)
az_cc_bbox <- st_bbox(coconino_plot)

az_cc_bbox <- c(
  left = az_cc_bbox["xmin"][[1]]-1,
  bottom = az_cc_bbox["ymin"][[1]]-1,
  right = az_cc_bbox["xmax"][[1]]+1,
  top = az_cc_bbox["ymax"][[1]]+1
)

coconino_map <- get_stadiamap(bbox = az_cc_bbox, zoom = 8)
```

```
## i © Stadia Maps © Stamen Design © OpenMapTiles © OpenStreetMap contributors.
```

```
# plot
ggmap(coconino_map) + labs(title = "Coconino County Naturally Caused Large Wildfire Incidence", subtitl
  geom_sf(data = coconino_plot, fill = "gray70", alpha = 0.5, color = "black", size = 5, inherit.aes =
  geom_sf(data=wf_sf_cc_plot, shape = 21, size = 0.9,
          color = "orangered3", fill = "orangered3", inherit.aes = FALSE) +
  geom_sf(data=bkg_sf_cc_plot, shape = 4, size = 1.1, color = "navyblue", inherit.aes = FALSE)
```

```
## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.
```



We first generate a realization of a Poisson point process to represent where Naturally occurring wildfires could have happened in Cocinino County, a simple features object called `background_sf`. For each of the points in the realization opted to generate all of the covariate data, including the census data for population density as well as all of the natural data. The data was a collaborative effort and required some tedious construction of useable dataframes that are column binded to the `background_sf` object. Those details are omitted and the data is provided as an `RData` file that can be imported.

The `background_sf` and `az_fires_wf_spglm`

```
df1 <- st_drop_geometry(az_fires_wf_spglm)
df2 <- st_drop_geometry(background_sf)


common_cols <- intersect(names(df1), names(df2))

df1 <- df1[, c(common_cols)]
df2 <- df2[, c(common_cols)]

df1$FireDiscoveryDateTime <- as.Date(df1$FireDiscoveryDateTime)
df2$FireDiscoveryDateTime <- as.Date(df2$FireDiscoveryDateTime)


Covariates <- rbind(df1, df2)

rm(df1,df2,df)

#### row bind to build a model_data_sf sf object for use in our model
```

```
all_points <- superimpose(unmark(az_fires_wf_spglm_ppp), background)

Wildfires <- c(rep(1, az_fires_wf_spglm_ppp$n), rep(0, background$n))

data <- cbind(Wildfires, Covariates)

model_data_sf <- st_as_sf(cbind(data, as.data.frame(all_points)[, c('x', 'y')]),
coords = c('x', 'y'))
```

A function to create categorical bins for season to use as predictors in wildfire incidence models.

```
get_season <- function(date) {
  month <- as.integer(format(date, "%m"))
  season <- case_when(
    month %in% c(12, 1, 2) ~ 1,
    month %in% c(3, 4, 5) ~ 2,
    month %in% c(6, 7, 8) ~ 3,
    month %in% c(9, 10, 11) ~ 4
  )
  return(season)
}
```

For our model selection, we used lowest AIC score to select a model. The below chunk produces the AIC for different variants of model fits. This approach is something like a "poor man's" version of the `stepAIC` function from the `MASS` package that can give a best-AIC model and do variable selection in an automated way. This way is not automated, and below is only serving as an example of how some models were compared. The chunk cycles through each model in `formula_list` and does a fit for each of the different `cov_types`. It fits each model using `spglm` and `binary` response and computes AIC for the model, returning a list from which we can select the best AIC model.

```
# Create a list of formula objects
formula_list <- list(
  as.formula("Wildfires ~ I(sqrt(distance_rd_min_prisec)) +
  I(sqrt(distance_rd_4wd)) + I(sqrt(distance_rd_min_isprisec)) +
  I(log(mean_slope)) + mean_forest + mean_grass + I(log(pop.density)) +
  Precipitation_Buffered + Temp_Min_Buffered + Temp_Max_Buffered +
  I(month(FireDiscoveryDateTime))"),
  as.formula("Wildfires ~  I(sqrt(distance_rd_min_isprisec)) +
  Precipitation_Buffered + Temp_Min_Buffered + I(get_season(FireDiscoveryDateTime)) +
  mean_grass * mean_forest"),
  as.formula("Wildfires ~  I(sqrt(distance_rd_min_isprisec)) + I(log(pop.density)) +
  Precipitation_Buffered * Temp_Min_Buffered +
  I(get_season(FireDiscoveryDateTime)) +
  mean_grass * mean_forest")

)

# Define spatial covariance types
# gaussian best for all so far
# OUT: cauchy and matern
cov_types <- c("wave",
               "gaussian",
               "spherical",
               "circular")
```

```r
# initialize results df
results_df <- data.frame(
  Model = character(),
  AIC = numeric(),
  Pseudo_R_squared = numeric(),
  stringsAsFactors = FALSE
)

# Outer loop for formulas
for (i in seq_along(formula_list)) {
  # Inner loop for spatial covariance types
  for (j in seq_along(cov_types)) {
    # Fit the model using spglm
    model <- spglm(formula_list[[i]], data = model_data_sf, family = binomial, spcov_initial = spcov_in

    model_summary <- summary(model)

    # build results df for each model
    new_row <- data.frame(
      Model = paste0("Model_", i, "_", cov_types[j]),
      AIC = AIC(model),
      Pseudo_R_squared = model_summary$pseudoR2,
      stringsAsFactors = FALSE
    )

    # Append the new row to the results dataframe
    results_df <- rbind(results_df, new_row)

  }
}

kable(results_df)
```

Model	AIC	Pseudo_R_squared
Model_1_wave	117.1134	0.5590062
Model_1_gaussian	365.1477	0.3586689
Model_1_spherical	365.2596	0.2998826
Model_1_circular	364.9773	0.3108486
Model_2_wave	341.4249	0.3670237
Model_2_gaussian	342.3983	0.3633874
Model_2_spherical	342.8299	0.3235570
Model_2_circular	342.6206	0.3283563
Model_3_wave	339.5729	0.3683819
Model_3_gaussian	340.4693	0.3684576
Model_3_spherical	340.7325	0.3265667
Model_3_circular	340.5777	0.3240059

This is only any example to show noteworthy results in this process.

Some models returned unusually low AICs. Upon investigating, these very-low AIC models were actually just models whose coefficients did not converge. These models are discarded, hence the model we ultimately used.

It's also interesting to note, as we will see below, that a predictor like `pop.density` can be added to the model

and improve it by a noticeable margin in terms of AIC score as well as R^2, but not be close to a level of significance that we would think it would be "useful" as a predictor, but in fact does improve the model by this measure.

Using lots of methodical trial and error (there were many trial/error steps that are omitted), we selected the model in the below chunk by lowest AIC.

Intercept Only Model (for reference)

```
# intercept-only model

spglm_formula <- Wildfires ~  1


az_wf_spcov <- spcov_initial("wave")

az_wf_spglm <- spglm(spglm_formula, data = model_data_sf,
                     family = binomial, spcov_initial = az_wf_spcov)

summary(az_wf_spglm)
```

```
##
## Call:
## spglm(formula = spglm_formula, family = binomial, data = model_data_sf,
##     spcov_initial = az_wf_spcov)
##
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1135 -0.4445 -0.1573  0.6972  2.6415
##
## Coefficients (fixed):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.4210     0.7767   -1.83   0.0673 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Coefficients (wave spatial covariance):
##        de        ie     range
## 9.127e+00 8.474e-03 2.090e+04
##
## Coefficients (Dispersion for binomial family):
## dispersion
##          1
```

```
kable(AIC(az_wf_spglm), caption = "Intercept-Only Model AIC")
```

Table 6: Intercept-Only Model AIC

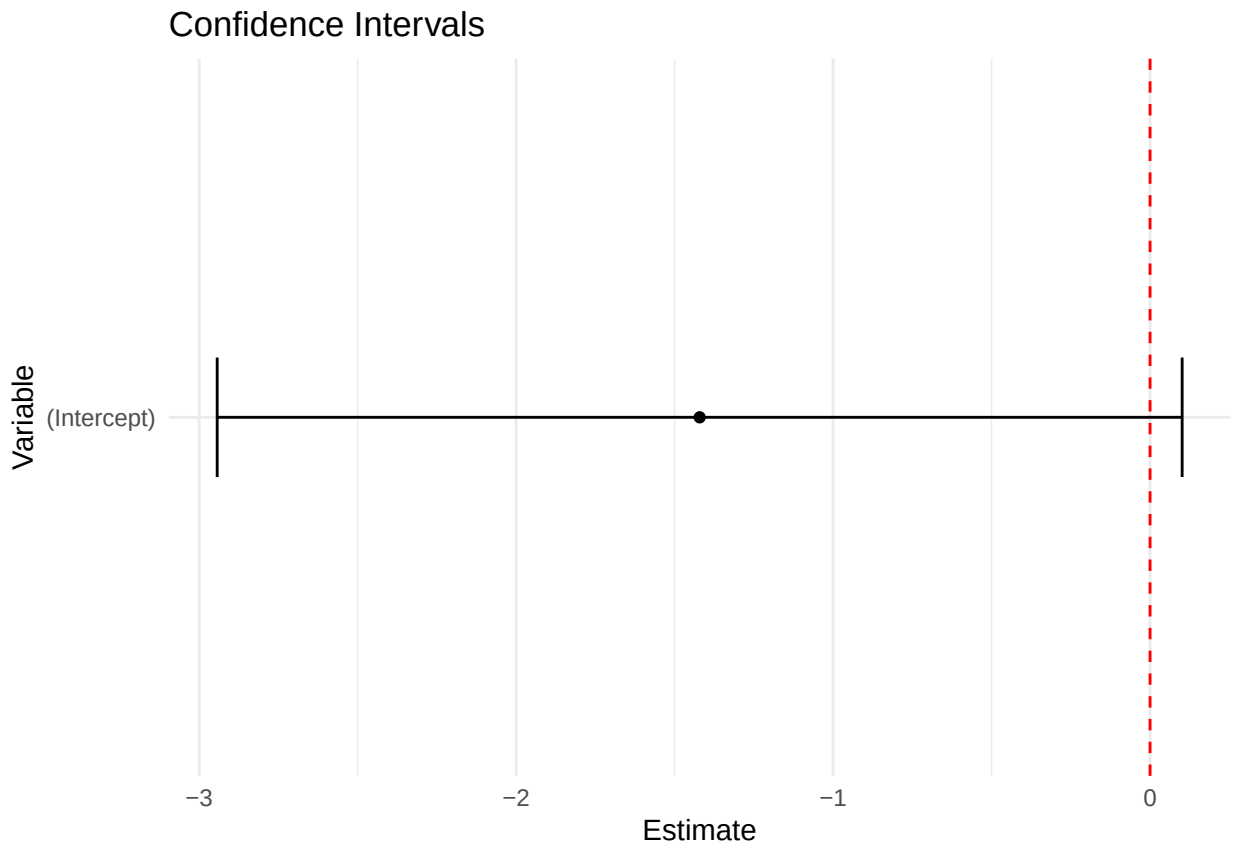
x
399.0993

```
az_wf_spglm_ci <- as.data.frame(confint(az_wf_spglm))

colnames(az_wf_spglm_ci) <- c("lower", "upper")

az_wf_spglm_ci$estimate <- az_wf_spglm$coefficients$fixed
```

```
az_wf_spglm_ci$variable <- rownames(az_wf_spglm_ci)

kable(az_wf_spglm_ci)
```

	lower	upper	estimate	variable
(Intercept)	-2.943177	0.1012612	-1.420958	(Intercept)

```
ggplot(az_wf_spglm_ci, aes(x = estimate, y = variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = lower, xmax = upper), height = 0.2) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Confidence Intervals",
       x = "Estimate",
       y = "Variable") +
  theme_minimal() +
  theme(axis.text.y = element_text(hjust = 0))
```



Best AIC model

```
# best AIC model

spglm_formula <- Wildfires ~  distance_rd_min_isprisec + I(log(pop.density)) +
  Precipitation_Buffered * Temp_Min_Buffered + I(get_season(FireDiscoveryDateTime)) +
  mean_grass * mean_forest
```

```r
az_wf_spcov <- spcov_initial("wave")

az_wf_spglm <- spglm(spglm_formula, data = model_data_sf,
                     family = binomial, spcov_initial = az_wf_spcov)

summary(az_wf_spglm)
```

```
##
## Call:
## spglm(formula = spglm_formula, family = binomial, data = model_data_sf,
##     spcov_initial = az_wf_spcov)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.79364 -0.37217 -0.03644  0.38603  2.67112
##
## Coefficients (fixed):
##                                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             -2.31282    7.91520  -0.292   0.7701
## distance_rd_min_isprisec                -1.32153    0.81237  -1.627   0.1038
## I(log(pop.density))                     -0.10851    0.49135  -0.221   0.8252
## Precipitation_Buffered                   0.06916    2.08641   0.033   0.9736
## Temp_Min_Buffered                       -1.12366    0.78317  -1.435   0.1514
## I(get_season(FireDiscoveryDateTime))     0.54990    0.37428   1.469   0.1418
## mean_grass                               1.07408    1.74733   0.615   0.5388
## mean_forest                              1.30509    1.21915   1.070   0.2844
## Precipitation_Buffered:Temp_Min_Buffered  0.39667    0.52455   0.756   0.4495
## mean_grass:mean_forest                  15.29506    9.16978   1.668   0.0953 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Pseudo R-squared: 0.3684
##
## Coefficients (wave spatial covariance):
##        de        ie     range
## 2.832e+00 3.549e-02 3.554e+04
##
## Coefficients (Dispersion for binomial family):
## dispersion
##          1
```

```r
kable(AIC(az_wf_spglm), caption = "Model AIC")
```

Table 8: Model AIC

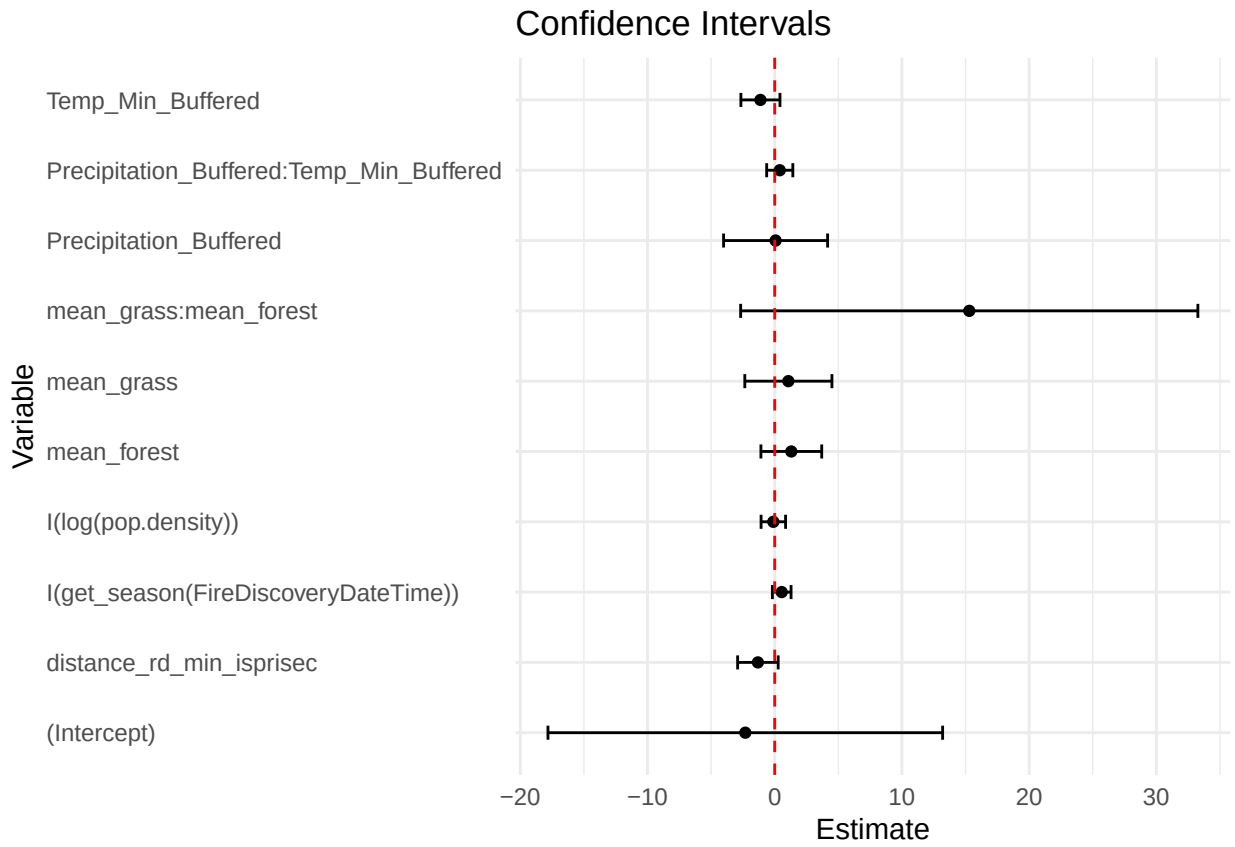
x
339.5729

```r
az_wf_spglm_ci <- as.data.frame(confint(az_wf_spglm))

colnames(az_wf_spglm_ci) <- c("lower", "upper")

az_wf_spglm_ci$estimate <- az_wf_spglm$coefficients$fixed
```

```r
az_wf_spglm_ci$variable <- rownames(az_wf_spglm_ci)

kable(az_wf_spglm_ci)
```

	lower	upper	estimate	variable
(Intercept)	-17.8263262	13.2006939	-2.3128162	(Intercept)
distance_rd_min_isprisec	-2.9137530	0.2706848	-1.3215341	distance_rd_min_isprisec
I(log(pop.density))	-1.0715342	0.8545142	-0.1085100	I(log(pop.density))
Precipitation_Buffered	-4.0201412	4.1584522	0.0691555	Precipitation_Buffered
Temp_Min_Buffered	-2.6586537	0.4113346	-1.1236596	Temp_Min_Buffered
I(get_season(FireDiscoveryDateTime))	-0.1836816	1.2834872	0.5499028	I(get_season(FireDiscoveryDateTime))
mean_grass	-2.3506191	4.4987742	1.0740775	mean_grass
mean_forest	-1.0844096	3.6945865	1.3050884	mean_forest
Precipitation_Buffered:Temp_Min_Buffered	-0.6314373	1.4247793	0.3966710	Precipitation_Buffered:Temp_Min_Buffered
mean_grass:mean_forest	-2.6773800	33.2675083	15.2950642	mean_grass:mean_forest

```r
ggplot(az_wf_spglm_ci, aes(x = estimate, y = variable)) +
  geom_point() +
  geom_errorbarh(aes(xmin = lower, xmax = upper), height = 0.2) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Confidence Intervals",
       x = "Estimate",
       y = "Variable") +
  theme_minimal() +
  theme(axis.text.y = element_text(hjust = 0))
```



To use our model for predictions, we had to weigh the challenges for acquiring data for our predictors with the time it takes to extract the data (some processing took a lot of time). Below is code that generates a grid of points in Coconino county, and gives us a total of 272 points evenly spaced across the county.

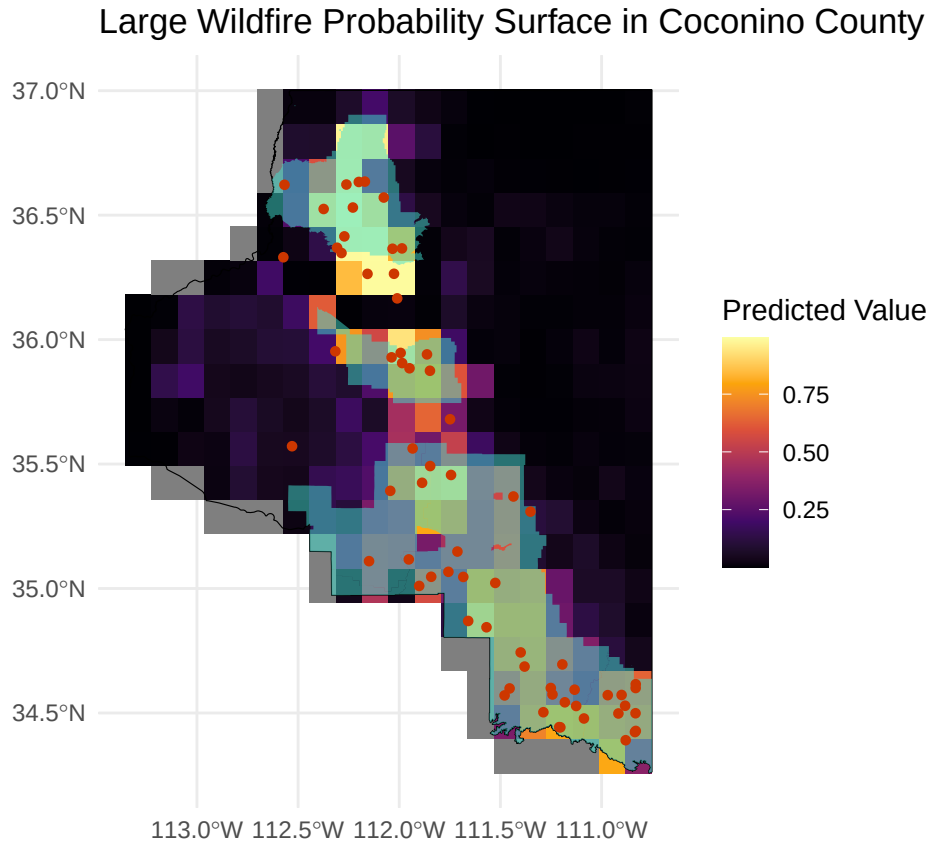
```
rm(az_wf_spglm_ci)
```

```
# Create a grid of points
grid <- st_make_grid(coconino_sf, n = c(20, 20), what = "centers")

# Convert the grid to an sf object
grid_sf <- st_sf(geometry = grid)

# Filter points to keep only those inside Coconino County
coconino_grid <- grid_sf[coconino_sf, ]

# If you need exactly 200 points, you can sample from the resulting grid
coconino_grid_n <- coconino_grid %>%
  slice_sample(n = 400)

# Plot to verify
plot(st_geometry(coconino_sf))
plot(st_geometry(coconino_grid_n), add = TRUE, col = "red", pch = 20)

prediction_grid <- cbind(st_drop_geometry(coconino_grid_n), st_coordinates(coconino_grid_n))

prediction_grid$FireDiscoveryDateTime <- ymd_hm("2023-12-31 12:00")
```

Predictions Once we have the points, we go through a similar process of generating the data for roads, census data, and natural factors. The natural factors were taken for the last day of 2023, so that the buffered temperature and precipitation data covers the year of 2023.

Using this data we can generate a plot of predictions from our model. The prediction values express probability that a large wildfire will occur if it were to start at the prediction point coordinates with the predictors used in the model (natural conditions, distances to roads), and population density for the captured data at those points. The visualization treats the prediction at that point as an approximation of other nearby points in the cell.

```
load(here('data/az_prediction_grid_sf.RData'))

az_wf_predict_spglm <- predict(az_wf_spglm, type = "response", se.fit = T, newdata = az_prediction_grid_

az_prediction_grid_sf$predict_bin_spglm <- az_wf_predict_spglm$fit

predict_plot <- st_transform(az_prediction_grid_sf, crs = 4326)

#national forest geometry for plots

naz_nf_plot <- st_transform(naz_forests_sf, crs = 4326 )


#generate pixel grid

plot_coconino_sf <- st_transform(coconino_sf, crs = 4326)

# Create a grid of squares
grid <- st_make_grid(plot_coconino_sf, n = c(20, 20), what = "polygons")

# Convert the grid to an sf object
grid_sf <- st_sf(geometry = grid)

# Filter squares to keep only those intersecting with Coconino County
coconino_grid <- grid_sf[plot_coconino_sf, ]

# Spatial join to transfer predicted values from points to grid squares
coconino_grid_with_values <- st_join(coconino_grid, predict_plot)

#plot

ggplot() +
  geom_sf(data = coconino_grid_with_values, aes(fill = predict_bin_spglm), color = NA) +
  geom_sf(data = coconino_sf, fill = NA, color = "black", size = 0.5) +
  geom_sf(data = naz_nf_plot, fill = "turquoise", alpha = 0.5, color = NA) +
  # geom_sf(data = coco_nf_plot, fill = "lawngreen", alpha = 0.5, color = NA) +
  # geom_sf(data = kaibab_nf_plot, fill = "green4", alpha = 0.5, color = NA) +
  geom_sf(data=wf_sf_cc_plot, shape = 21, size = 1.2,
          color = "orangered3", fill = "orangered3", inherit.aes = FALSE) +
  scale_fill_viridis_c(option = "B") +
  theme_minimal() +
  labs(title = "Large Wildfire Probability Surface in Coconino County",
       fill = "Predicted Value")
```



Since natural factors like temperature and precipitation can vary year to year, we can modify these predictors to see what kind of effects it has on our predictions.

Predictor Sweep Animations *HTML only*