Evaluation of Population Health and Economic Damage caused by Storms and Severe Weather

girldewhirle

Monday, May 11, 2015

Synopsis

This analysis is to assess the impact, both in terms of human fatalities/injuries and economic damage, that extreme weather events cause in the US. The data for this analysis was obtained from U.S. National Oceanic and Atmospheric Administration's (NOAA) storm database. The analysis focuses on answering two questions:

- Across the United States, which types of events (as indicated in the EVTYPE variable) are most harmful with respect to population health?
- Across the United States, which types of events have the greatest economic consequences?

Data Processing

Find latest date and then do last ten years.

Load libraries required for data cleanse and analysis

```
## load libraries - warnings suppressed for this chunk only
library("plyr")
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("tidyr")
library("lubridate")
```

```
##
## Attaching package: 'lubridate'
##
## The following object is masked from 'package:plyr':
##
##     here
```

```
library("knitr")
library("stringr")
library("ggplot2")
library("reshape2")
library("lattice")
library("gridExtra")
```

```
## Loading required package: grid
```

Raw data was accessed from https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2

The transformation of the code starts from the presumption that raw data is already in working directory. The data is read into workspace as follows.

```
## read in data
storm_data<-read.csv("Stormdata",header=TRUE)
## convert BGN_Date to date formate
storm_data$BGN_DATE<-mdy_hms(storm_data$BGN_DATE)
```

It is believed that there are fewer records for earlier years and that later years have better reporting of severe weather events. Reviewing the frequency of the instances of reported events in the data over time it is possible to see that the number of events reported increases significantly over time. Therefore, to give more consistent results, a ten year period from the most recent date is used as the data set to analyse.

To identify the most recent event date the entire data set was sorted by the BGN_DATE variable. The most recent date is then picked from the top the date 10 years ago identified.

```
storm_data<-arrange(storm_data,desc(BGN_DATE))
## identify most recent date
most_recent_date<-storm_data[1,2]
## 10 years ago
ten_years_ago<-most_recent_date - years(10)
```

The data set was then subsetted into those events that occurred within the last ten years.

```
storm_data<-filter(storm_data,(BGN_DATE>=ten_years_ago)&
                     (BGN_DATE<=most_recent_date))
```

For analysis, to better answer the individual questions, the data was split into two main subsets: events causing human fatalities and/or injuries, and events causing economic damage.

Human Impact

Human impact records were subsetted on the basis that either they caused greater than zero fatalities OR they caused greater than zero injuries.

```
## filter based on fatalities or injuries != 0
human<-filter(storm_data,FATALITIES!=0|INJURIES!=0)
```

Economic Impact

Economic impact (damage) records were subsetted on the basis that either the PROPDMG (property damage) field was greater than zero OR that the CROPDMG (crop damage) field was greater than zero.

```
## create data set where property or crop damage != 0
economic<-filter(storm_data,PROPDMG!=0|CROPDMG!=0)
```

Creation of Helper Functions

A number of helper functions were created to run against the EVTYPE data to standardise the values.

A variable of the 48 standard values from the [National Weather Service Storm Data Documentation] (https://d396qusza40orc.cloudfront.net/repdata%2Fpeer2_doc%2Fpd01016005curr.pdf)

This was then used in a helper function (check_if)std) during analysis to identify progress in standardising values. This function is not required to reproduce this analysis - it was run during original exploratory work to identify if/when EVTYPE values had been standardised - but it is included here for sake of completion.

```
## create variable of standard event values from documentation
standard_events<-c("Astronomical Low Tide","Avalanche","Blizzard",
                   "Coastal Flood","Cold/Wind Chill","Debris Flow","Dense Fog",
                   "Dense Smoke","Drought","Dust Devil","Dust Storm",
                   "Excessive Heat","Extreme Cold/Wind Chill",
                   "Flash Flood","Flood","Frost/Freeze","Funnel Cloud"
                   ,"Freezing Fog","Hail","Heat","Heavy Rain","Heavy Snow",
                   "High Surf","High Wind","Hurricane (Typhoon)","Ice Storm",
                   "Lake-Effect Snow","Lakeshore Flood","Lightning",
                   "Marine Hail","Marine High Wind","Marine Strong Wind",
                   "Marine Thunderstorm Wind","Rip Current","Seiche","Sleet",
                   "Storm Surge/Tide","Strong Wind","Thunderstorm Wind",
                   "Tornado","Tropical Depression","Tropical Storm","Tsunami",
                   "Volcanic Ash","Waterspout","Wildfire","Winter Storm",
                   "Winter Weather")
## convert standards to upper case
standard_events<-toupper(standard_events)
## create helper function to check if values are standardised sucessfully
check_if_std<-function(event_type){
  out<-"not std"
  if (event_type %in% standard_events){
    out<-"std"
    return(out)
  }
  return(out)
}
```

The functions below were written individually based on one or more standard values so that the order they are applied to the different data subsets can be varied and, if some of the standard values do not exist in any subset then the function is not required at all.

```
## create number of helper functions to clean event types
flood<-function(event_type){
  out<-event_type
  flood_check<-(grepl("flood",event_type,ignore.case=TRUE)|
                  grepl(" fld",event_type,ignore.case=TRUE))&!
    (grepl("coastal",event_type,ignore.case=TRUE)|
       grepl("flash",event_type,ignore.case=TRUE)|
       grepl("cstl",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
```

```
    out<-"Flood"
    return(out)
  }
  return(out)
}

heavy_rain<-function(event_type){
  out<-event_type
  check<-grepl("rain",event_type,ignore.case=TRUE)|
    grepl("wetness",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Heavy Rain"
    return(out)
  }
  return(out)
}

coastal_flood<-function(event_type){
  out<-event_type
  check<-grepl("flood",event_type,ignore.case=TRUE)&
    (grepl("coastal",event_type,ignore.case=TRUE)|
       grepl("cstl",event_type,ignore.case=TRUE))
  if (check==TRUE){
    out<-"Coastal Flood"
    return(out)
  }
  return(out)
}

flash_flood<-function(event_type){
  out<-event_type
  check<-(grepl("flood",event_type,ignore.case=TRUE)|
            grepl("fld",event_type,ignore.case=TRUE))&
    grepl("flash",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Flash Flood"
    return(out)
  }
  return(out)
}

snow<-function(event_type){
  out<-event_type
  flood_check<-(grepl("snow",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Heavy Snow"
    return(out)
  }

  return(out)
}

high_wind<-function(event_type){
```

```
  out<-event_type
  flood_check<-(grepl("high wind",event_type,ignore.case=TRUE)|
                  grepl("gusty",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"High Wind"
    return(out)
  }

  return(out)
}

rip_current<-function(event_type){
  out<-event_type
  flood_check<-(grepl("rip current",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Rip Current"
    return(out)
  }

  return(out)
}


thunderstorm_wind<-function(event_type){
  out<-event_type
  flood_check<-((grepl("thunderstorm wind",event_type,ignore.case=TRUE)|
                   grepl("TSTM wind",event_type,ignore.case=TRUE)|
                   grepl("thunder",event_type,ignore.case=TRUE)|
                   grepl("microburst",event_type,ignore.case=TRUE))&!
                  grepl("marine",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Thunderstorm Wind"
    return(out)
  }

  return(out)
}

marine_thunderstorm_wind<-function(event_type){
  out<-event_type
  flood_check<-((grepl("thunderstorm wind",event_type,ignore.case=TRUE)|
                   grepl("TSTM wind",event_type,ignore.case=TRUE)|
                   grepl("thunder",event_type,ignore.case=TRUE))&
                  grepl("marine",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Marine Thunderstorm Wind"
    return(out)
  }

  return(out)
}

tornado<-function(event_type){
```

```
  out<-event_type
  flood_check<-(grepl("tornado",event_type,ignore.case=TRUE)&!
                  grepl("marine",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Tornado"
    return(out)
  }

  return(out)
}

lightning<-function(event_type){
  out<-event_type
  flood_check<-(grepl("lightning",event_type,ignore.case=TRUE)&!
                  grepl("marine",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Lightning"
    return(out)
  }

  return(out)
}

excessive_heat<-function(event_type){
  out<-event_type
  flood_check<-(grepl(" heat",event_type,ignore.case=TRUE))|
    grepl("heat wave",event_type,ignore.case=TRUE)
  if (flood_check==TRUE){
    out<-"Excessive Heat"
    return(out)
  }
  return(out)
}

wildfire<-function(event_type){
  out<-event_type
  flood_check<-(grepl("wild",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Wildfire"
    return(out)
  }
  return(out)
}

landslide_debris_flow<-function(event_type){
  out<-event_type
  flood_check<-(grepl("slide",event_type,ignore.case=TRUE))|
    (grepl("debris",event_type,ignore.case=TRUE)&
       grepl("flow",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    out<-"Debris Flow"
    return(out)
  }
```

```
  return(out)
}

## for extreme cold also including hypothermia
extreme_cold<-function(event_type){
  out<-event_type
  flood_check<-(grepl("cold",event_type,ignore.case=TRUE))|
    grepl("wind chill",event_type,ignore.case=TRUE)|
    grepl("hypoth",event_type,ignore.case=TRUE)
  if (flood_check==TRUE){
    out<-"Extreme Cold/Wind Chill"
    return(out)
  }
  return(out)
}

fog<-function(event_type){
  out<-event_type
  flood_check<-(grepl("fog",event_type,ignore.case=TRUE))
  if (flood_check==TRUE){
    freeze<-grepl("freeze",event_type,ignore.case=TRUE)
    if (freeze==TRUE){
      out<-"Freezing Fog"
      return(out)
    }
    out<-"Dense Fog"
    return(out)
  }
  return(out)
}

winter_weather<-function(event_type){
  out<-event_type
  check<-grepl("winter",event_type,ignore.case=TRUE)&
    grepl("weather",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Winter Weather"
    return(out)
  }
  return(out)
}

surf<-function(event_type){
  out<-event_type
  check<-grepl("surf",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"High Surf"
    return(out)
  }
  return(out)
}

surge<-function(event_type){
```

```
  out<-event_type
  check<-grepl("surge",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"STORM SURGE/TIDE"
    return(out)
  }
  return(out)
}

hail<-function(event_type){
  out<-event_type
  check<-grepl("hail",event_type,ignore.case=TRUE)&!
    grepl("marine",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Hail"
    return(out)
  }
  return(out)
}

hurricane<-function(event_type){
  out<-event_type
  check<-grepl("cane",event_type,ignore.case=TRUE)|
    grepl("phoon",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Hurricane (Typhoon)"
    return(out)
  }
  return(out)
}

frost_freeze<-function(event_type){
  out<-event_type
  check<-grepl("freeze",event_type,ignore.case=TRUE)|
    grepl("frost",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Frost/Freeze"
    return(out)
  }
  return(out)
}

tropical_storm<-function(event_type){
  out<-event_type
  check<-grepl("tropical",event_type,ignore.case=TRUE)&
    grepl("storm",event_type,ignore.case=TRUE)
  if (check==TRUE){
    out<-"Tropical Storm"
    return(out)
  }
  return(out)
}
```

The function below is to convert the exponential values in the PROPDMGEXP and CROPDMGEXP fields

to a numerical value. For the values in those variables, if they did not equal K, M or B (or k, m or b) the multiplier was returned as 1.

```
## function to convert damage values to multiplier
convert_exp<-function(value){
  out<-1
  value<-as.character(value)
  if (value=="K"|value=="k"){
    out<-1000
    return(out)
  }
  if (value=="M"|value=="m"){
    out<-1000000
    return(out)
  }
  if (value=="B"|value=="b"){
    out<-1000000000
    return(out)
  }
  return(out)
}
```

Application of EVTYPE standards

For human data the number of fatalities and injuries per event were added together to give a total value per event for human impact. Weighting of fatalities was considered so that events with the same total human impact but with higher fatalities were considered to have more of an impact. However, this was not carried out as it was deemed arbitary and would distort the results.

The cutoff value for the top 1% of total human impact was calculated using the quantile function. The human data was then subset into events where the total human impact was greater than the 1% cut off to give the top 1% of events by human impact. This was done to effectively be able to answer the question 'Across the United States, which types of events (as indicated in the EVTYPE variable) are most harmful with respect to population health?' the top 1% of events by total impact were considered.

For economic data the exponential values were converted to numerical values (K=1,000, M=1,000,000, B=1,000,000,000) and then used to multiply their respective property and crop damage values. These numerical values were then added together to give a total amount of economic damage for each event. The cutoff value for the top 1% of total economic impact was calculated using the quantile function. The economic data was then subset into events where the total economic impact was greater than the 1% cut off to give the top 1% of events by economic impact. This was done to be able to effectively answer the question 'Across the United States, which types of events have the greatest economic consequences?' by considering the events which cause the most economic impact.

From the two 'top 1%' subsets - human and economic - tables of unique event types and their frequencies were extracted.

The helper functions were run on these tables in order of event frequency until all listed events had a matching standardised value. This was done on the human data and the economic data seperately.

When all the values had been standardised (a helper function was used to run against the event types as they were cleansed to ascertain if they met the standards or not) the event mapping tables were tidied and then merged back into their respective 'top 1%' tables.

The 'top 1%' tables with the standard event types were used as the clean data for analysis.

Summary tables for human and economic impact were then created using the summarise function and used for presentation below.

```
## create subset only including event type, number of fatalities, number of
## injuries and reference number
minimal_human<-select(human,EVTYPE,BGN_DATE,FATALITIES,INJURIES,REFNUM)
## create variable which is a result of fatalities + injuries
minimal_human<-mutate(minimal_human,fat_plus_inj=FATALITIES+INJURIES)
## find value for top 1%
top_1_per_cutoff<-quantile(minimal_human$fat_plus_inj,prob=0.99)
## subset data to top 1% max human impact
top_human_impact<-filter(minimal_human,fat_plus_inj>top_1_per_cutoff)
## create table of unique values with frequency to be basis of mapping table
unique_top_human<-as.data.frame(table(top_human_impact$EVTYPE))
## filter out zero frequencies
unique_top_human<-filter(unique_top_human,Freq>0)
## sort in descending order as will start to apply helper functions based on
## most frequent in top 1%
unique_top_human<-arrange(unique_top_human,desc(Freq))
## convert Var1 to character
unique_top_human$Var1<-as.character(unique_top_human$Var1)

## start cleaning in following order
##tornado, excessive heat, flood, heat (not run), flash flood, hurricane/typhoon
## at this point checked for remaining none standard
## values - only extreme cold and wildfire left
unique_top_human<-mutate(unique_top_human,ev2=mapply(tornado,Var1))%>%
  mutate(ev3=mapply(excessive_heat,ev2))%>%
  mutate(ev4=mapply(flood,ev3))%>%
  mutate(ev5=mapply(flash_flood,ev4))%>%
  mutate(ev6=mapply(hurricane,ev5))%>%
  mutate(ev7=mapply(extreme_cold,ev6))%>%
  mutate(ev8=mapply(wildfire,ev7))
## resulting table is now merge table so crop down to use
unique_top_human<-select(unique_top_human,Var1,ev8)
## rename to allow join
colnames(unique_top_human)[1]<-"EVTYPE"
colnames(unique_top_human)[2]<-"cleaned_EVTYPE"
## join with top % human impact file to get cleaned values
minimal_cleaned_human<-join(top_human_impact,unique_top_human)
```

```
## Joining by: EVTYPE
```

```
## trim table
minimal_cleaned_human<-select(minimal_cleaned_human,cleaned_EVTYPE,
                              FATALITIES:REFNUM,fat_plus_inj)
## grouped by event type
grouped_ev_type<-group_by(minimal_cleaned_human,cleaned_EVTYPE)
## summarise and get mean fatalities and mean injuries and also total(sum) values
## as integer for clarity
Mean_Human_Impact<-summarise(grouped_ev_type,as.integer(mean(FATALITIES)),
                             as.integer(mean(INJURIES)),as.integer
                             (sum(FATALITIES)),as.integer(sum(INJURIES)),
                             as.integer(mean(fat_plus_inj)),
                             as.integer(sum(fat_plus_inj)))
```

```
## rename columns
colnames(Mean_Human_Impact)[2]<-"Mean_Fatalities"
colnames(Mean_Human_Impact)[3]<-"Mean_Injuries"
colnames(Mean_Human_Impact)[4]<-"Total_Fatalities"
colnames(Mean_Human_Impact)[5]<-"Total_Injuries"
colnames(Mean_Human_Impact)[6]<-"Mean_Human_Impact"
colnames(Mean_Human_Impact)[7]<-"Total_Human_Impact"
## arrange in desc order fatalities
Mean_Human_Impact<-arrange(Mean_Human_Impact,desc(Total_Human_Impact))
```

```
## create subset of economic with only event type, damage values and refnum
minimal_economic<-select(economic,EVTYPE,BGN_DATE,PROPDMG,
                         PROPDMGEXP,CROPDMG,CROPDMGEXP,REFNUM)
## convert exp values to a numeric multiplier
minimal_economic<-mutate(minimal_economic,
                         prop_multi=mapply(convert_exp,PROPDMGEXP),
                         crop_multi=mapply(convert_exp,CROPDMGEXP))
## calculate damage for property and crops
minimal_economic<-mutate(minimal_economic,prop_damage=PROPDMG*prop_multi,
                         crop_damage=CROPDMG*crop_multi)
## combine property and crop damage
minimal_economic<-mutate(minimal_economic,total_damage=prop_damage+crop_damage)
## identify top 1% cut off
top_damage<-quantile(minimal_economic$total_damage,prob=0.99)
## subset based on greater than cutoff
economic_damage<-filter(minimal_economic,total_damage>top_damage)
## create merge table based on unique values
unique_economic_top<-as.data.frame(table(economic_damage$EVTYPE))
## remove zero occurances
unique_economic_top<-filter(unique_economic_top,Freq>0)
## sort by Freq
unique_economic_top<-arrange(unique_economic_top,desc(Freq))
## convert Var1 to character
unique_economic_top$Var1<-as.character(unique_economic_top$Var1)
## start applying helper functions in order - tornado, flood, flash flood, hail,
## drought(already standard), ice storm(already standard), tstm wind,
## hurricane, wildfire, extreme cold/wind chill,landslide, high winds,
## coastal flooding
unique_economic_top<-mutate(unique_economic_top,ev2=mapply(tornado,Var1))%>%
  mutate(ev3=mapply(flood,ev2))%>%
  mutate(ev4=mapply(flash_flood,ev3))%>%
  mutate(ev5=mapply(hail,ev4))%>%
  mutate(ev6=mapply(thunderstorm_wind,ev5))%>%
  mutate(ev7=mapply(hurricane,ev6))%>%
  mutate(ev8=mapply(wildfire,ev7))%>%
  mutate(ev9=mapply(extreme_cold,ev8))%>%
  mutate(ev10=mapply(landslide_debris_flow,ev9))%>%
  mutate(ev11=mapply(high_wind,ev10))%>%
  mutate(ev12=mapply(surge,ev11))%>%
  mutate(ev13=mapply(frost_freeze,ev12))%>%
  mutate(ev14=mapply(coastal_flood,ev13))%>%
  mutate(ev15=mapply(heavy_rain,ev14))%>%
  mutate(ev16=mapply(tropical_storm,ev15))
```

```
## create table with original and revised values
economic_merge<-select(unique_economic_top,Var1,ev16)
## rename column 1
colnames(economic_merge)[1]<-"EVTYPE"
colnames(economic_merge)[2]<-"new_event"
## join with economic damage data set
economic_table<-join(economic_damage,economic_merge)
```

```
## Joining by: EVTYPE
```

```
## remove original event type
economic_table<-select(economic_table,new_event,BGN_DATE,PROPDMG:total_damage)
## group by event type
economic_table<-group_by(economic_table,new_event)
## create summary table
economic_summary<-summarise(economic_table,sum(prop_damage),mean(prop_damage),
                            sum(crop_damage),mean(crop_damage),
                            sum(total_damage),mean(total_damage))
## rename columns
colnames(economic_summary)[2]<-"Sum_Prop_DMG"
colnames(economic_summary)[3]<-"Mean_Prop_DMG"
colnames(economic_summary)[4]<-"Sum_Crop_DMG"
colnames(economic_summary)[5]<-"Mean_Crop_DMG"
colnames(economic_summary)[6]<-"Sum_Total_DMG"
colnames(economic_summary)[7]<-"Mean_Total_DMG"
## arrange by sum total damage
economic_summary<-arrange(economic_summary,desc(Sum_Total_DMG))
```

The mapping table for cleaning the event types in the human data is:

```
kable(arrange(unique_top_human,EVTYPE),
      caption="Merge Table for Cleaning EVTYPE for Human Data")
```

EVTYPE	cleaned_EVTYPE
EXCESSIVE HEAT	Excessive Heat
FLASH FLOOD	Flash Flood
FLOOD	Flood
HAIL	HAIL
HEAT	HEAT
HEAVY SNOW	HEAVY SNOW
HIGH WIND	HIGH WIND
HURRICANE/TYPHOON	Hurricane (Typhoon)
THUNDERSTORM WIND	THUNDERSTORM WIND
TORNADO	Tornado
TROPICAL STORM	TROPICAL STORM
TSUNAMI	TSUNAMI

EVTYPE	cleaned_EVTYPE
WILDFIRE	Wildfire
WINTER WEATHER	WINTER WEATHER

Table 1: Merge Table for Cleaning EVTYPE for Human Data

The mapping table for cleaning the event types in the economic data is:

```
kable(arrange(economic_merge,EVTYPE),
      caption="Merge Table for Cleaning EVTYPE for Economic Data")
```

EVTYPE	new_event
BLIZZARD	BLIZZARD
COASTAL FLOOD	Coastal Flood
DROUGHT	DROUGHT
EXCESSIVE HEAT	EXCESSIVE HEAT
EXTREME COLD	Extreme Cold/Wind Chill
FLASH FLOOD	Flash Flood
FLOOD	Flood
FROST/FREEZE	Frost/Freeze
HAIL	Hail
HEAVY RAIN	Heavy Rain
HEAVY SNOW	HEAVY SNOW
HIGH SURF	HIGH SURF
HIGH WIND	High Wind
HURRICANE	Hurricane (Typhoon)
HURRICANE/TYPHOON	Hurricane (Typhoon)
ICE STORM	ICE STORM
LAKE-EFFECT SNOW	LAKE-EFFECT SNOW
LANDSLIDE	Debris Flow
LIGHTNING	LIGHTNING
STORM SURGE	STORM SURGE/TIDE
STORM SURGE/TIDE	STORM SURGE/TIDE
STRONG WIND	STRONG WIND
THUNDERSTORM WIND	Thunderstorm Wind
TORNADO	Tornado
TROPICAL STORM	Tropical Storm
TSTM WIND	Thunderstorm Wind

EVTYPE	new_event
TSTM WIND/HAIL	Hail
TSUNAMI	TSUNAMI
WILD/FOREST FIRE	Wildfire
WILDFIRE	Wildfire
WINTER STORM	WINTER STORM
WINTER WEATHER	WINTER WEATHER

Table 2: Merge Table for Cleaning EVTYPE for Economic Data

```
## melt data to allow for multi series plots
human_top_melted<-melt(top_human_impact,id=c("EVTYPE","BGN_DATE","REFNUM"))
## convert Mean_Human_Impact to data frame and then melt
Mean_Human_Impact<-as.data.frame(Mean_Human_Impact)
## melt summary table
melted_human_impact<-melt(Mean_Human_Impact,id.vars="cleaned_EVTYPE")
```

```
## melt data to allow for multi series plots
## need to convert to data frame first
economic_table<-as.data.frame(economic_table)
## melt data
melted_economic_table<-melt(economic_table,
                            id.vars=c("new_event","BGN_DATE","REFNUM"))
```

```
## Warning: attributes are not identical across measure variables; they will
## be dropped
```

```
## convert economic summary to data frame
economic_summary<-as.data.frame(economic_summary)
## melt data
melted_economic_summary<-melt(economic_summary,id.vars = "new_event")
```

Results

Using cleaned data tables containing the events that cause the most human and economic damage.

```
## display table
kable(Mean_Human_Impact)
```

cleaned_EVTYPE	Mean_Fatalities	Mean_Injuries	Total_Fatalities	Total_Injuries	Mean_Human_Impa
Tornado	11	178	401	6418	1
Excessive Heat	6	154	89	2006	1
Hurricane (Typhoon)	7	400	23	1200	4
HEAT	4	160	25	963	1

cleaned_EVTYPE	Mean_Fatalities	Mean_Injuries	Total_Fatalities	Total_Injuries	Mean_Human_Impa
TROPICAL STORM	1	200	1	200	2
Wildfire	11	75	22	151	
HAIL	0	81	0	163	
TSUNAMI	32	129	32	129	1
WINTER WEATHER	1	137	1	137	1
Flash Flood	0	136	0	136	1
HEAVY SNOW	0	100	0	100	1
Flood	1	92	1	92	
HIGH WIND	0	70	0	70	
THUNDERSTORM WIND	0	70	0	70	

```
kable(economic_summary)
```

new_event	Sum_Prop_DMG	Mean_Prop_DMG	Sum_Crop_DMG	Mean_Crop_DMG	Sum_To
Flood	130427809100	460875651	3228548000	1.140830e+07	133
Hurricane (Typhoon)	72214530000	1223975085	3028070000	5.132322e+07	75
STORM SURGE/TIDE	47676900000	5297433333	0	0.000000e+00	47
Tornado	15278620000	72755333	116915000	5.567381e+05	15
Flash Flood	7710191350	43073695	546871000	3.055145e+06	8
Hail	7683309000	69848264	509590000	4.632636e+06	8
DROUGHT	832664000	18924182	5734820000	1.303368e+08	6
Wildfire	4481680000	91462857	270703000	5.524551e+06	4
High Wind	4096380000	89051739	446628000	9.709304e+06	4
Thunderstorm Wind	2748300000	59745652	277350000	6.029348e+06	3
Tropical Storm	1805180000	53093529	371660000	1.093118e+07	2
ICE STORM	1689860000	51207879	8550000	2.590909e+05	1
WINTER STORM	1033950000	68930000	0	0.000000e+00	1
Frost/Freeze	90000	6000	958490000	6.389933e+07	
Heavy Rain	322000000	16947368	290540000	1.529158e+07	
EXCESSIVE HEAT	170000	170000	492400000	4.924000e+08	
Debris Flow	232800000	23280000	20000000	2.000000e+06	
Coastal Flood	170480000	24354286	0	0.000000e+00	
STRONG WIND	91750000	22937500	63400000	1.585000e+07	
TSUNAMI	121800000	40600000	20000	6.666667e+03	
Extreme Cold/Wind Chill	0	0	65050000	6.505000e+07	
BLIZZARD	62000000	62000000	0	0.000000e+00	

new_event	Sum_Prop_DMG	Mean_Prop_DMG	Sum_Crop_DMG	Mean_Crop_DMG	Sum_To
HEAVY SNOW	61250000	15312500	0	0.000000e+00	
HIGH SURF	47920000	23960000	0	0.000000e+00	
LIGHTNING	29000000	14500000	0	0.000000e+00	
LAKE-EFFECT SNOW	20000000	20000000	0	0.000000e+00	
WINTER WEATHER	0	0	15000000	1.500000e+07	

```
## plot of total human impact by event type
human_plot_object<-ggplot(data=Mean_Human_Impact,
                          aes(cleaned_EVTYPE,Total_Human_Impact))
## create object to plot as bar with text changed to read easier
hpo<-human_plot_object + geom_bar(stat="identity",colour="green",fill="blue") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  xlab("Severe Weather Event Type")+ylab("Total Human Impact")+
  ggtitle("Total Human Damage by Weather Event Type")
```

```
## plot of total economic damage by event type
## economic plot object
economic_plot_object<-ggplot(data=economic_summary,aes(new_event,Sum_Total_DMG))
## create object to plot as bar with text changed to read easier
epo<-economic_plot_object +
  geom_bar(stat="identity",colour="red",fill="yellow") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  xlab("Severe Weather Event Type") + ylab("Sum of Total Economic Damage") +
  ggtitle("Total Economic Damage by Weather Event Type")
```

Figure One: Total Human Impact by Weather Event Types

```
hpo
```

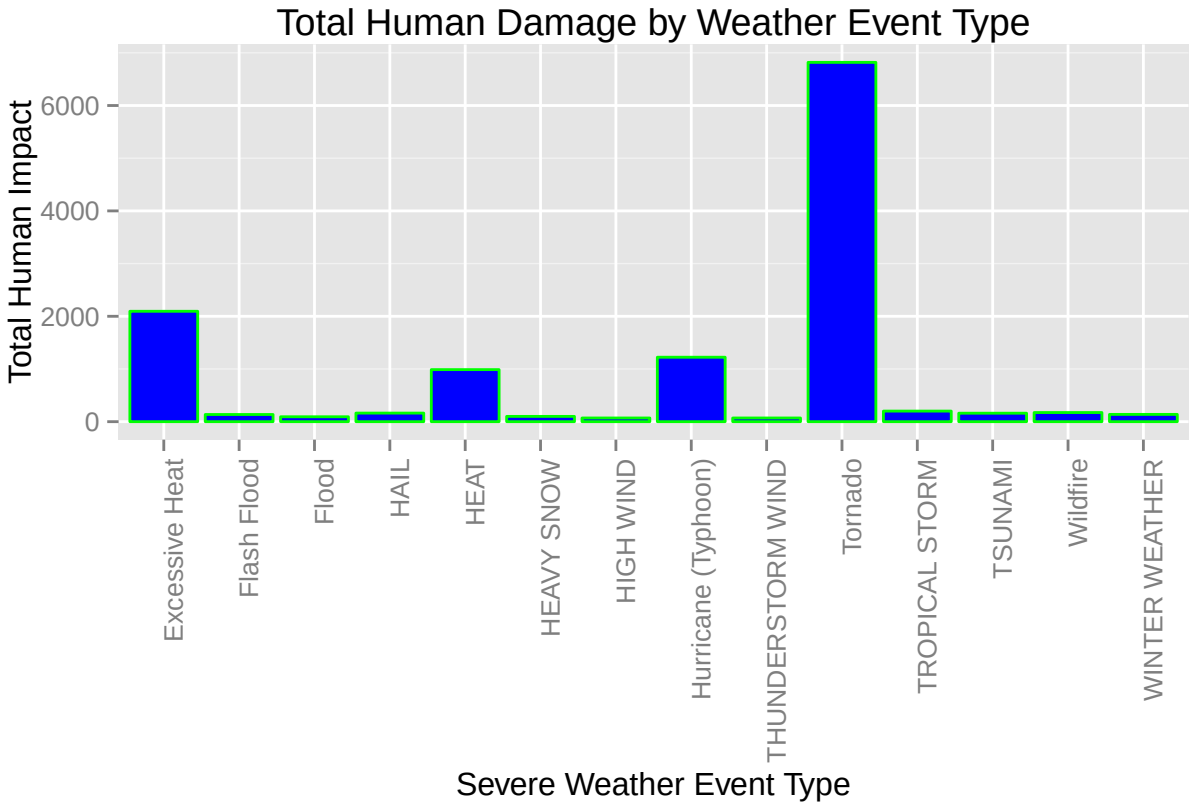
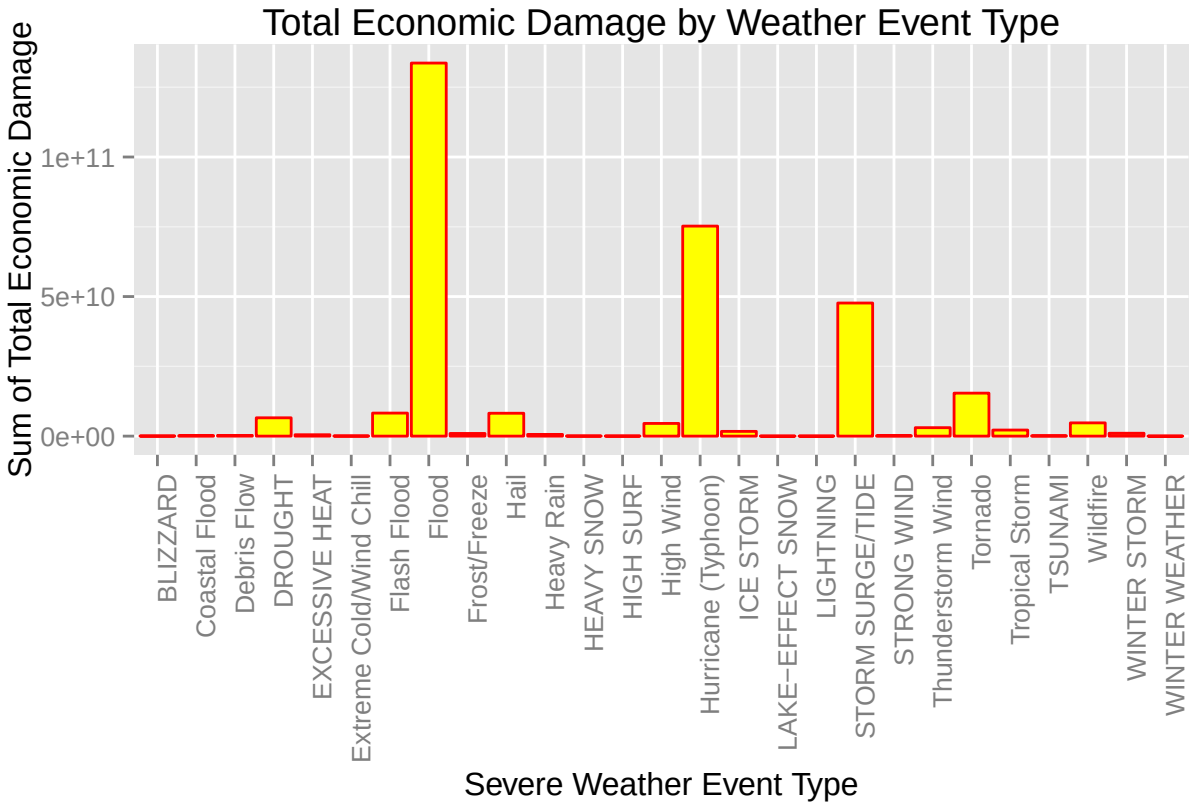


Figure Two: Total Economic Impact by Weather Event Types

```
epo
```



Summary

On the basis of the analysis the following conclusions have been drawn.

The following event types cause the most human impact (measured by total injuries and fatalities for all instances of the event type in the human data subset):

- Tornado
- Excessive Heat
- Hurricane (Typhoon)

The following event types cause the most economic impact (measured by total property and crop damage for all instances of the event type in the economic data subset):

- Flood
- Hurricane (Typhoon)
- Storm Surge/Tide