NOAA data analysis : Impact of storms on Health and Economy

Rithvik Shamrao Vinekar

2023-03-02

Synopsis

The National Oceanic and Atmospheric Administration department of the US government collects data about storms and atmospheric events from across the country. Specific people submit reports of the events which are then collated into reports. These reports are made available as datasets for researchers. One such dataset is provided at https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2.There is linked documentation which tells us the format expected as well as the well-defined categories of events, and much more information.

The databases provide information about property damages, crop damages, fatalities and injuries reported, among other things. It also includes a Remark, which has a verbal description of the event, as well as the location of the event reported. This includes state and county information, time of event and the latitude and longitude of the event. The categories of events involve zones, such as County level or Marine events.

The questions we need to answer are:

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

Ideally, a thorough analysis of such a dataset would involve a thorough reading of the Remarks in order to categorise the events. It is often difficult to give a single category to an event as many phenomena can occur simultaneously, but the number of fatalities, injuries or damages refers to a single event. The location data can also tell if an event is marine, coastal or a land event. Such a thorough study is beyond scope of this document.

In this document, the provided Event data will be used. The early datasets did not have a fixed format of entry or did not enforce it. Thus the people who reported events have made entries which are inconsistent and have many human errors. The major part of the analysis involves cleaning and categorising the data into the events expected in the documentation, and a few more which may help.

Subsequently the data can be easily grouped by event and values summed, in this case Fatalities and Injuries for Health, and Property Damage and Crop Damage for Economy. For the latter, the values are in multiples of 1, K (1000), M(million) or B(billion), so some processing needs to be done to get them in a single unit - USD $. We will attempt to categorise the whole data, from 1950 to 2011, as requested.

Data Processing

This is the session info for the R session used currently:

Session info

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31)
## Platform: aarch64-apple-darwin20 (64-bit)
## Running under: macOS Ventura 13.2
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2-arm64/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] compiler_4.2.2  fastmap_1.1.0   cli_3.6.0       tools_4.2.2
##  [5] htmltools_0.5.4 rstudioapi_0.14 yaml_2.3.7      rmarkdown_2.20
##  [9] knitr_1.42      xfun_0.36       digest_0.6.31   rlang_1.0.6
## [13] evaluate_0.20
```

Files are available on Github

Obtaining the data

Loading required libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.1
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.3.0      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(sjmisc)
```

```
##
## Attaching package: 'sjmisc'
##
## The following object is masked from 'package:purrr':
##
##     is_empty
##
## The following object is masked from 'package:tidyr':
##
##     replace_na
##
## The following object is masked from 'package:tibble':
##
##     add_case
```

Download the data.

```
url_source<-"https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2"
file_dest <- "StormData.csv.bz2"
download.file(url_source,file_dest, method="libcurl")
rm( url_source, file_dest)
```

Read the compressed csv file directly into a dataframe : `stormdata`

```
stormdata <- read.csv("StormData.csv.bz2", encoding = "UTF-8")
```

Check the data

The data read into `stormdata` needs to be checked, starting with str()

```
str(stormdata)
```

```
## 'data.frame':    902297 obs. of  37 variables:
##  $ STATE__   : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ BGN_DATE  : chr  "4/18/1950 0:00:00" "4/18/1950 0:00:00" "2/20/1951 0:00:00" "6/8/1951 0:00:00" .
##  $ BGN_TIME  : chr  "0130" "0145" "1600" "0900" ...
##  $ TIME_ZONE : chr  "CST" "CST" "CST" "CST" ...
##  $ COUNTY    : num  97 3 57 89 43 77 9 123 125 57 ...
##  $ COUNTYNAME: chr  "MOBILE" "BALDWIN" "FAYETTE" "MADISON" ...
##  $ STATE     : chr  "AL" "AL" "AL" "AL" ...
##  $ EVTYPE    : chr  "TORNADO" "TORNADO" "TORNADO" "TORNADO" ...
##  $ BGN_RANGE : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ BGN_AZI   : chr  "" "" "" "" ...
##  $ BGN_LOCATI: chr  "" "" "" "" ...
##  $ END_DATE  : chr  "" "" "" "" ...
##  $ END_TIME  : chr  "" "" "" "" ...
##  $ COUNTY_END: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ COUNTYENDN: logi  NA NA NA NA NA NA ...
##  $ END_RANGE : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ END_AZI   : chr  "" "" "" "" ...
##  $ END_LOCATI: chr  "" "" "" "" ...
##  $ LENGTH    : num  14 2 0.1 0 0 1.5 1.5 0 3.3 2.3 ...
##  $ WIDTH     : num  100 150 123 100 150 177 33 33 100 100 ...
##  $ F         : int  3 2 2 2 2 2 2 1 3 3 ...
##  $ MAG       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ FATALITIES: num  0 0 0 0 0 0 0 0 1 0 ...
##  $ INJURIES  : num  15 0 2 2 2 6 1 0 14 0 ...
##  $ PROPDMG   : num  25 2.5 25 2.5 2.5 2.5 2.5 2.5 25 25 ...
##  $ PROPDMGEXP: chr  "K" "K" "K" "K" ...
##  $ CROPDMG   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ CROPDMGEXP: chr  "" "" "" "" ...
##  $ WFO       : chr  "" "" "" "" ...
##  $ STATEOFFIC: chr  "" "" "" "" ...
##  $ ZONENAMES : chr  "" "" "" "" ...
##  $ LATITUDE  : num  3040 3042 3340 3458 3412 ...
##  $ LONGITUDE : num  8812 8755 8742 8626 8642 ...
##  $ LATITUDE_E: num  3051 0 0 0 0 ...
##  $ LONGITUDE_: num  8806 0 0 0 0 ...
##  $ REMARKS   : chr  "" "" "" "" ...
##  $ REFNUM    : num  1 2 3 4 5 6 7 8 9 10 ...
```

There are 902297 entries

The questions needed to answer are again:

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

The two questions involve comparing EVTYPE with

1. FATALITIES and INJURIES
2. PROPDMG and CROPDMG

In the latter case, there is a need to get damage expenses in a numerical format in the same unit. This is currently not so, as PROPDMGEXP and CROPDMGEXP hold unit multipliers like "",”K”, “M”,“B” which stand basically for 1, 1,000, 1,000,000, 1,000,000,000 units or none, Kilo, Million and Billion. This info is available in linked documentation

Before any of this is done, check EVTYPE if it is correct. According to linked documentation (Table 1, Page 6), there are finite number of types. This data is available on the pdf file, under the heading

2.1.1 Storm Data Event Table

These types were put in a text file from this documentation by copy and paste. To make it a csv file, a grep search&replace using `/\s([ZCM])$/,\1/` to replace the space separating Z or C or M at the end of the line with a comma was done. This file has been saved as `"Stormdata_EvType.csv"` with contents that look like below:-

```
Astronomical Low Tide,Z
Avalanche,Z
Blizzard,Z
Coastal Flood,Z
Cold/Wind Chill,Z
Debris Flow,C
Dense Fog,Z
Dense Smoke,Z
Drought,Z
Dust Devil,C
Dust Storm,Z
...
```

This was read into a variable dataframe stormevents from the csv file

```
stormevents <- read.csv("Stormdata_EvType.csv", header = FALSE)
str(stormevents)
```

```
## 'data.frame':    48 obs. of  2 variables:
##  $ V1: chr  "Astronomical Low Tide" "Avalanche" "Blizzard" "Coastal Flood" ...
##  $ V2: chr  "Z" "Z" "Z" "Z" ...
```

Give the stormevents dataframe proper header names

```
names(stormevents)<-c("Event_name","Designator")
stormevents$Event_name<-toupper(stormevents$Event_name)
stormevents =  stormevents %>%
    arrange(desc(str_length(Event_name)), Event_name)
```

Now we need to check if the events in the EVTYPE entry match these

```
length(unique(stormdata$EVTYPE))
```

```
## [1] 985
```

There are 985 unique entries instead of 48.

```
uniqueEvTypes <- data.frame(EVTYPE=unique(stormdata$EVTYPE))
head(uniqueEvTypes)
```

```
##                    EVTYPE
## 1                 TORNADO
## 2               TSTM WIND
## 3                    HAIL
## 4           FREEZING RAIN
## 5                    SNOW
## 6 ICE STORM/FLASH FLOOD
```

One can also use `View(uniqueEvTypes)` to look at all entries interactively in RStudio. Looking at these entries, there are many spelling errors, Summary entries which do not match events, and entries with different cases.

There are many EVTYPEs which are synonymous, but will differ due to spelling mistakes, variations in writing or spelling, use of synonymous words, and the understanding of the reporting person. Due to this, aggregating totals and sums from such data is not possible unless all related EVTYPES are binned together.

Since there are so many spurious summary "events", we can add one more category or bin

```
stormevents[nrow(stormevents) + 1,] <- c("SUMMARY","S")
```

It may not be possible to automate every instance and change them to match the categories we need. Some may be irrelevant, such as the summary entries. However, we can go a long way to do so.

The ideal way to do this is to have boolean columns for each category and map each event to category based on whether the event fits the category or not. This is because each event can, and most probably do belong to multiple categories. Tornadoes, Hurricanes etc. can also secondarily fall into Strong Wind, Storm Surge, Heavy Rain, Lightening etc. and many other such categories. Events do not occur in isolation, but data for crop and property damage do not get separated into these categories easily. However, such an approach is beyond the scope of this work.

Preparation

Save EVTYPE into a column ORIG_EVTYPE. This way EVTYPE can be used as a working copy, while ORIG_EVTYPE preserves original values, if needed.

```
stormdata$ORIG_EVTYPE <- stormdata$EVTYPE
```

UPPERCASE Events

Now all the EVTYPE are made Uppercase. This eliminates many entries with inconsistent cases.

```
stormdata$EVTYPE <- toupper(stormdata$EVTYPE)
```

Common replacements

We can replace some common word occurrances with substitutions. These substitutions are from observations that some words are synonyms or other ways to express the same thing, like High temperature to mean high heat,etc. The list below is not exhaustive, but is continually updated. It could be put into a file or variable, but separate commands are currently used.

Note that there will be an automatic process of word matching used later, so these replacements aid the automatic process later to categorise events.

```
stormdata$EVTYPE<-gsub("PRECIPITATION","RAIN" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("PRECIP","RAIN" , stormdata$EVTYPE)
```

```
# Precipitation can mean rain, snow, hail, any and all of these mixed up, but rain is most common.
stormdata$EVTYPE<-gsub("TSTM.*$","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("TSTM WIND","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("THUNDERSTORM WIN.*$","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HURRICANE.*$","HURRICANE (TYPHOON)",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FREEZING RAIN","SLEET",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("GLAZE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("SNOW/ICE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("SNOW","SLEET",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FREEZING DRIZZLE","SLEET",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HEAVY MIX","SLEET",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COOL","COLD", stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COLD WAVE","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("(HEAT WAVE|HOT SPELL)","EXCESSIVE HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("UNSEASONABLY HEAT AND DRY","EXCESSIVE HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("UNSEASONABLY DRY","DROUGHT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("UNSEASONABLY WET","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("SMOKE","DENSE SMOKE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RAIN","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WARM","HEAT", stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH TEMPERATURE","EXCESSIVE HEAT" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("UNSEASONABLY HOT","EXCESSIVE HEAT" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LOW TEMPERATURE","COLD/WIND CHILL" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXTREME HEAT","EXCESSIVE HEAT" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RECORD COLD AND HIGH WIND","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
## Spelling mistake Torndao instead of Tornado :-)
stormdata$EVTYPE<-gsub("TORNDAO","TORNADO",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("DUST DEVEL","DUST DEVIL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WIND CHILL","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXTREME/RECORD COLD","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RECORD COLD EXTREME","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXCESSIVE COLD","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXTREME COLD","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COLD$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RECORD RAINFALL","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXCESSIVE/EXCESSIVE RAINFALL","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("(MIXED|EXCESSIVE|MONTHLY) RAIN","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXTREME/RECORD COLD","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("UNSEASONAL LOW TEMP","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RECORD","EXCESSIVE" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXCESSIVE TEMPERATURE","EXCESSIVE HEAT" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("TEMPERATURE EXCESSIVE","EXCESSIVE HEAT" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("SEVERE","EXCESSIVE" , stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COLD$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("THUNDERSTORM","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LIGHTING","LIGHTNING",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LIGHTENING","LIGHTNING",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LIGNTNING","LIGHTNING",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*HIGH TIDE.*","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH WAVES","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WIND DAMAGE","STRONG WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH WIND/SEAS","MARINE HIGH WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*WALL CLOUD","TROPICAL STORM",stormdata$EVTYPE)
```

```
stormdata$EVTYPE<-gsub("EXCESSIVE HIGH","EXCESSIVE HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXCESSIVE LOW","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WIND","HIGH WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("STORM SURGE","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FOG","DENSE FOG",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FUNNEL","FUNNEL CLOUD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("AVALANCE","AVALANCHE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^ICE$","SLEET",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^FREEZE$","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FROST.*FREEZE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("VOLCANIC.*","VOLCANIC ASH",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*FROST$","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH SURF","HEAVY SURF",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WILD FIRES","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WILD FIRE","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WILD/FOREST FIRE","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WATER SPOUT","WATERSPOUT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WAYTERSPOUT","WATERSPOUT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("BLOWING DUST","DUST STORM",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RAINSTORM","TROPICAL STORM",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RAIN STORM","TROPICAL STORM",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("DAMAGING FREEZE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("RAINFALL","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("SHOWER","RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("(MUDSLIDES|MUDSLIDE|MUD/ROCK SLIDE)","DEBRIS FLOW",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LANDSLIDE","DEBRIS FLOW",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH SEAS","MARINE THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("TYPHOON","HURRICANE (TYPHOON)",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("MICROBURST","TORNADO",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("GUSTNADO","TORNADO",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("DOWNBURST","TORNADO",stormdata$EVTYPE)
# downbursts and microbursts are different from tornadoes, but similar enough
stormdata$EVTYPE<-gsub("HEAVY SURF","HIGH SURF",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("ROUGH SEAS","MARINE THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HEAVY SEAS","MARINE THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH SEAS","MARINE THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COASTALFLOOD","COASTAL FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("CSTL FLOOD","COASTAL FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EROSION","COASTAL FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HYPOTHERMIA/EXPOSURE","EXTREME COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COLD WEATHER","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HIGH WATER","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("(HIGH|HEAVY) SWELLS","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("MARINE (MISHAP|ACCIDENT)","MARINE THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("URBAN/SML STREAM FLD","FLASH FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*STREAM.*FL.*D","FLASH FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FLASH FLOO*D","FLASH FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("WINT.?R.? MIX","WINTER WEATHER",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("ICY ROAD.*","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("BLACK ICE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("AGRICULTURAL FREEZE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HARD FREEZE","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("ROUGH SURF","HIGH SURF",stormdata$EVTYPE)
```

```
stormdata$EVTYPE<-gsub("ROGUE WAVE","HIGH SURF",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("BRUSH FIRE","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FOREST FIRE","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("EXCESSIVE COLD/FROST","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COASTAL.?STORM","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("COASTAL.?SURGE","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("APACHE COUNTY","STRONG WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("METRO STORM.*","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^EXCESSIVE$","EXCESSIVE HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^RED FLAG CRITERIA$","LIGHTNING",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^HIGH$","HIGH WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^MONTHLY TEMPERATURE$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("REMNANTS OF FLOYD","HURRICANE (TYPHOON)",stormdata$EVTYPE)
## Hurricane Floyd - in case you are wondering :-)
stormdata$EVTYPE<-gsub("VOG","VOLCANIC ASH",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^ICE$","SLEET",stormdata$EVTYPE)

## Mainly in Hawaii, volcanic ash mixed into fog
stormdata$EVTYPE<-gsub("RAPIDLY RISING WATER","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*WET","HEAVY RAIN",stormdata$EVTYPE)
## Remarks say this entry is not a flash flood, just heavy rain which caught a man cleaning a manhole o
```

For EVTYPE given as OTHER, we can try something different

Now, we can check the uniqueEvTypes again

```
uniqueEvTypes <- data.frame(EVTYPE_M=unique(stormdata$EVTYPE))
dim(uniqueEvTypes)
```

```
## [1] 663   1
```

Already there is a reduction in number of unique entries, from 985 to 663, but this is not enough. We will need to match the entries using text matching. One way of doing so is string distance, but it is not very accurate on its own.

Another would be to have separate columns for each of the 48 variables as mentioned before, and check if the unique values have anything to do with them. The evtypes can sometimes map to multiple events, such as "RIP CURRENTS HIGH SURF" which maps to both RIP CURRENTS and HIGH SURF. This would lead to a lengthy detailed analysis, however for this short analysis we will assume each evtype maps to one of the 48 variables.

Note that all changes must be done on the main stormdata, not on uniqueEvTypes, because otherwise the mappings will be lost. We will have to finally regenerate and merge `uniqueEvTypes` with `stormdata`, after it has matching columns from `stormevents`

First we will have a function which can find how many matching words are present. This checks if the evtype we are looking for is a substring in the entry.

```
getstr_contain<-function(i, data = stormdata){
  x=""
  numstormevents <- dim(stormevents)[1]
  check = 0
  for(j in 1:numstormevents) {
      if (str_contains(data$EVTYPE[i],stormevents$Event_name[j]) > 0) {
        x <- stormevents$Event_name[j]
        return(x)
      }
  }

 return(x)
}
```

Now, we compare word to word

```
words_common <- function(str1, str2) {
  str1 <- toupper(str1)
  str2 <- toupper(str2)
  mapply(function(x, y) sum(str_contains(x, y)),
         strsplit(str1, ' '), strsplit(str2, ' '))
}

get_commonwords <- function(i, data = stormdata){
    x=""
    for(j in 1:dim(stormevents)[1]) {
      if (words_common(data$EVTYPE[i],stormevents$Event_name[j]) == TRUE) {
      x <- stormevents$Event_name[j]
      return(x)
      }
    }
    x
}
```

Finally, we use a fuzzy matching algorthm routinely used for our purpose, agrep. It was found that agrep is not really as accurate in matching as required, so it has lowest priority. These three filters are applied in order, thus filling as many gaps as possible.

```
compare_agrep<-function(i,data = stormdata){
  x<-agrep(data$EVTYPE[i], stormevents$Event_name,
  ignore.case=TRUE, value=TRUE,
  max.distance = 0.01, useBytes = TRUE)
 x <- paste0(x,"")
 x
}
```

Now we apply the filters defined in the above functions. uniqueEvTypes is regenerated, just in case to avoid any mismapping later. uniqueEvTypes$EVTYPE must not change in the analysis below. A new column EVTYPE_M is generated.

```
uniqueEvTypes <- data.frame(EVTYPE=unique(stormdata$EVTYPE))
for(i in 1:dim(uniqueEvTypes)[1]) {

 x <- getstr_contain(i,data=uniqueEvTypes)
 y <- get_commonwords(i,data=uniqueEvTypes)
 #z <- compare_agrep(i)

 if(length(x)>0) { uniqueEvTypes$EVTYPE_M[i] <- x }
 else if(length(y)>0) { uniqueEvTypes$EVTYPE_M[i] <- y }
# else if(length(z)>0) { uniqueEvTypes$EVTYPE_M[i] <- z }
 else { uniqueEvTypes$EVTYPE_M[i]<-NA }
}
```

```
stormdata <- merge(stormdata,uniqueEvTypes, by.x = "EVTYPE", by.y = "EVTYPE")
```

With this merger, work has to be done to stormdata, not uniqueEvTypes. We will now filter EVTYPE more with smaller terms. If this is done before, it can override large term detection and introduce bias.

Below are catch-alls with small words and wildcards like .*. They can mess with other, more specific entries. For example the first one will ensure even DUST DEVIL are DUST STORM so we need phase 2 replacement, after specific entries have been already detected They have therefore been moved after one run of the replace for loop. The loop will be run again for entries which are not filled in.

```
stormdata$EVTYPE<-gsub(".*DUST.*","DUST STORM",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*SURF","HIGH SURF",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*SWELL.*","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*SLIDE","DEBRIS FLOW",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*EROSI.?N","DEBRIS FLOW",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*DRY.*","DROUGHT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*TIDE.*","STORM SURGE/TIDE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*DAM.*","FLASH FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("ICE JAM","FLASH FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LANDSPOUT","TORNADO",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("LANDSLUMP","DEBRIS FLOW",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*WET.*","HEAVY RAIN",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("DROWNING","FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*DRIE.*","DROUGHT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("FREEZING SPRAY","FREEZING FOG",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*FREEZ.*","FROST/FREEZE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*HOT.*","HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*FIRE.*","WILDFIRE",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*TURBULENCE.*","
STRONG WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*ICE.*","ICE STORM",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HYPERTHERMIA/EXPOSURE","HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("HYPOTHERMIA.*","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*SMALL STREAM.*","FLOOD",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub(".*WND.*","STRONG WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("REMNANTS OF FLOYD","HURRICANE (TYPHOON)",stormdata$EVTYPE)
## Hurricane Floyd - in case you are wondering :-)
stormdata$EVTYPE<-gsub("VOG","VOLCANIC ASH",stormdata$EVTYPE)
## Mainly in Hawaii, volcanic ash mixed into fog
stormdata$EVTYPE<-gsub("RAPIDLY RISING WATER","HEAVY RAIN",stormdata$EVTYPE)
## Remarks say this entry is not a flash flood, just heavy rain which caught a man cleaning a manhole o
stormdata$EVTYPE<-gsub("APACHE COUNTY","STRONG WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("METRO STORM.*","THUNDERSTORM WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^EXCESSIVE$","EXCESSIVE HEAT",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^RED FLAG CRITERIA$","LIGHTNING",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^HIGH$","HIGH WIND",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^MONTHLY TEMPERATURE$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^COLD TEMPERATURE$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^COLD TEMPERATURES$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^COLD SPELL$","COLD/WIND CHILL",stormdata$EVTYPE)
stormdata$EVTYPE<-gsub("^URBAN.*","FLASH FLOOD",stormdata$EVTYPE)
```

Now the replacement loop is run again, this time using stormdata as the database. For the "OTHER" entries, which have registered values (Property and Crop damage or fatalities), we can run our detection on the REMARKS. This will ensure that "OTHER" entries also get a place.

```r
#uniqueEvTypes <- data.frame(EVTYPE=unique(stormdata$EVTYPE))
getstr_contain_remarks<-function(i, data = stormdata){
    x=""
    numstormevents <- dim(stormevents)[1]
    check = 0
    for(j in 1:numstormevents) {
        if (str_contains(data$REMARKS[i],stormevents$Event_name[j],ignore.case = TRUE) > 0) {
            x <- stormevents$Event_name[j]
            return(x)
        }
    }
    return(x)
}
get_commonwords_frm_remarks <- function(i, data = stormdata){
    x=""
    for(j in 1:dim(stormevents)[1]) {
        if (words_common(data$REMARKS[i],stormevents$Event_name[j]) == TRUE) {
            x <- stormevents$Event_name[j]
            return(x)
        }
    }
    x
}
```

The main function

```r
for(i in 1:dim(stormdata)[1]) {
    if (  stormdata$EVTYPE_M[i]=="" | is.na(stormdata$EVTYPE_M[i])){
        ## This is the second run, so we skip entries made before and ensure
        ## they don't get overwritten. This is the difference with first run
        if(stormdata$EVTYPE[i]=="OTHER"){
            x <- getstr_contain_remarks(i)
            y <- get_commonwords_frm_remarks(i)
            z <- ""
        } else {
            x <- getstr_contain(i)
            y <- get_commonwords(i)
            z <- compare_agrep(i)
            z <- z[length(z)]
        }
        if(length(x)>0 & x != "") { stormdata$EVTYPE_M[i] <- x }
        else if(length(y)>0 & y != "") { stormdata$EVTYPE_M[i] <- y }
        else if(length(z)>0 & z != "") { stormdata$EVTYPE_M[i] <- z }
    }
}
```

The unassigned values can be checked in a variable

```r
unassigned <- stormdata[stormdata$EVTYPE_M=="",]
```

Use the above variable unassigned, check for remnant unclassified and read the remarks. This gives a clue to what the remaining ones are. Examples are “REMNANTS OF FLOYD” which would leave us guessing what the remnant of Sgt Floyd at the Sioux City Museum has to do with the weather. The remarks say hurricane Floyd. The replacements code above is updated until the unassigned number is low and there are no more assignments. This is an iterative process, but once done, all replacements have been mentioned. Now, let the unassigned ones get a category of their own "OTHER". This leaves a handful - around 4 entries which can really not be categorised, such as NORTHERN LIGHTS, NONE, MILD etc. which are just observations or dummy entries. We can put all these entries into "OTHER"

```
stormdata[stormdata$EVTYPE_M=="",]$EVTYPE_M="OTHER"
```

The EVTYPE_M column now should be made factors

```
stormdata$EVTYPE_M <- as.factor(stormdata$EVTYPE_M)
```

Finally, we have our EVTYPE variables in the proper format as EVTYPE_M

Damage Values

There are two types of damage values : Property and Crop damage. Both should be expressed in USD or $, but they have been expressed in multiples. There is a PROPDMG column with a value, and a PROPDMGEXP column which specifies a code for a multiplier. These are supposed to be letters like K, M, B which are Kilo, Million, Billion - this much is mentioned in documentation. However

```
table(stormdata$PROPDMGEXP)
```

```
##
##              -      ?      +      0      1      2      3      4      5      6
## 465934       1      8      5    216     25     13      4      4     28      4
##      7       8      B      h      H      K      m      M
##      5       1     40      1      6 424665      7  11330
```

This shows that there are more entries than K,M,B.

What are H, 1..8 and -,+,? Instead of hazarding a guess, one way is to look for documentation from NOAA website. This does not help, as the format seems to have changed to a more logical simpler format. However, the discussion forum pointed the way. It is difficult to find the exact format as the formats from NOAA data have changed post 2012, according to the link from the discussion forum. Without this, the merge step below was removing rows.

First, we need to get all unique entries, which were mentioned in the table() output above. This is for PROPDMGEXP, there is also the CROPDMGEXP column. We need to get unique entries from these columns, make a union of the two sets, and sort them out.

```
unique_DMGEXP <- sort( union(
                  unique(stormdata$PROPDMGEXP),
                  unique(stormdata$CROPDMGEXP)
                            )
                  )
unique_DMGEXP
```

```
##  [1] ""  "-" "?" "+" "0" "1" "2" "3" "4" "5" "6" "7" "8" "B" "h" "H" "k" "K" "m"
## [20] "M"
```

Now we need to make a translation table which we will later merge. There is a csv mentioned on the discussion forum, but it is convenient to make the entries here itself.

```
Expense_units <- data.frame(DMGEXP = unique_DMGEXP,DMGUNIT = c(0,1,0,1,10,10,10,10,10,10,10,10,10,10000
```

Now we need to merge the expense units with the database twice, once for PROPDMG and once for CROPDMG

```
stormdata <- merge(stormdata,Expense_units,by.x="PROPDMGEXP",by.y="DMGEXP")
stormdata <- stormdata %>%
              rename("PROPDMGUNIT" = "DMGUNIT" )
```

```
stormdata <- merge(stormdata,Expense_units,by.x="CROPDMGEXP",by.y="DMGEXP")
stormdata <- stormdata %>%
              rename("CROPDMGUNIT" = "DMGUNIT" )
```

Now we calculate two new columns, which is what we want, the actual property and crop damage numbers in a single unit - USD or $.

```
stormdata <- stormdata %>%
              mutate( PROPDMGVAL= as.numeric(PROPDMG) * PROPDMGUNIT,                    CROPDMGVAL=
                    )
```

The columns we are interested in now are EVTYPE_M vs FATALITIES, INJURIES and CROPDMGVAL,PROPDMGVAL

Results

Population Health

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

The EVTYPE variable has now been changed to EVTYPE_M, which has been binned to the recommended 48 categories as well as a couple of extras (SUMMARY and OTHER). Some are missing as they were not detected:

```
unique(stormdata$EVTYPE_M)
```

```
##  [1] HAIL                     FLASH FLOOD              FLOOD
##  [4] EXCESSIVE HEAT           FUNNEL CLOUD             DENSE FOG
##  [7] BLIZZARD                 HEAVY RAIN               SLEET
## [10] HIGH WIND                DROUGHT                  COASTAL FLOOD
## [13] DEBRIS FLOW              HURRICANE (TYPHOON)      DUST DEVIL
## [16] AVALANCHE                DUST STORM               STORM SURGE/TIDE
## [19] DENSE SMOKE              TORNADO                  COLD/WIND CHILL
## [22] LIGHTNING                FROST/FREEZE             WATERSPOUT
## [25] HEAT                     HIGH SURF                WINTER STORM
## [28] EXTREME COLD/WIND CHILL  ICE STORM                MARINE HAIL
## [31] MARINE THUNDERSTORM WIND WILDFIRE                 RIP CURRENT
## [34] SEICHE                   SUMMARY                  TROPICAL STORM
## [37] LAKE-EFFECT SNOW         WINTER WEATHER           TROPICAL DEPRESSION
## [40] VOLCANIC ASH             OTHER                    MARINE STRONG WIND
## [43] STRONG WIND              THUNDERSTORM WIND        TSUNAMI
## [46] ASTRONOMICAL LOW TIDE    LAKESHORE FLOOD
## 47 Levels: ASTRONOMICAL LOW TIDE AVALANCHE BLIZZARD ... WINTER WEATHER
```

```
health <- stormdata %>%
              group_by(EVTYPE_M) %>%
              summarise(Fatalities=sum(FATALITIES, na.rm=TRUE),
                        Injuries=sum(INJURIES,na.rm=TRUE))
```

```
health_grp <- gather(health,Incident, Totals, Fatalities:Injuries, factor_key=TRUE, na.rm = TRUE)
```

The plot of the fatalities and injuries for each event are seen below

```
gg <- ggplot(data = health_grp, aes(x = EVTYPE_M, y = log10(Totals), fill=Incident))+
        geom_col()+
        ggtitle("Affect on Health")+
        xlab("Event") + ylab("Count log10")+
        facet_grid(Incident~.)+
        theme(plot.title = element_text(hjust = 0.5),
              axis.text.x = element_text(angle = 90, vjust = 0,
              hjust=0),axis.text=element_text(size=8)) +
        scale_y_continuous(breaks=c(0,1,2,3,4,5),
                        labels=c("0","10","100","1K","10K","100K"))
      gg
```

Cause of Maximum fatalities

The following events are most harmful - with fatalities ranked above injuries : These have caused greater than 1K fatalities:

```
health %>% arrange( desc(Fatalities)) %>% filter(Fatalities > 1000)
```

```
## # A tibble: 4 x 3
##   EVTYPE_M       Fatalities Injuries
##   <fct>               <dbl>    <dbl>
## 1 TORNADO              5636    91392
## 2 EXCESSIVE HEAT       2201     7124
## 3 HIGH WIND            1660    11803
## 4 FLASH FLOOD          1064     1881
```

Cause of Maximum injuries

These events have left a large number of injured:

```
health %>% arrange( desc(Injuries)) %>% filter(Injuries > 1000)
```

```
## # A tibble: 14 x 3
##    EVTYPE_M            Fatalities Injuries
##    <fct>                    <dbl>    <dbl>
##  1 TORNADO                   5636    91392
##  2 HIGH WIND                 1660    11803
##  3 EXCESSIVE HEAT            2201     7124
##  4 FLOOD                      483     6795
##  5 LIGHTNING                  817     5231
##  6 HEAT                       978     2119
##  7 ICE STORM                   90     1976
##  8 FLASH FLOOD               1064     1881
##  9 WILDFIRE                    90     1608
## 10 HAIL                        15     1371
## 11 WINTER STORM               217     1353
## 12 HURRICANE (TYPHOON)        135     1333
## 13 SLEET                      175     1284
## 14 DENSE FOG                   81     1077
```

Damages to Crops and Property

Across the United States, which types of events have the greatest economic consequences?

```
damages <- stormdata %>%
              group_by(EVTYPE_M) %>%
              summarise(Property=sum(PROPDMGVAL, na.rm = TRUE),
                        Crops=sum(CROPDMGVAL,na.rm = TRUE))
```

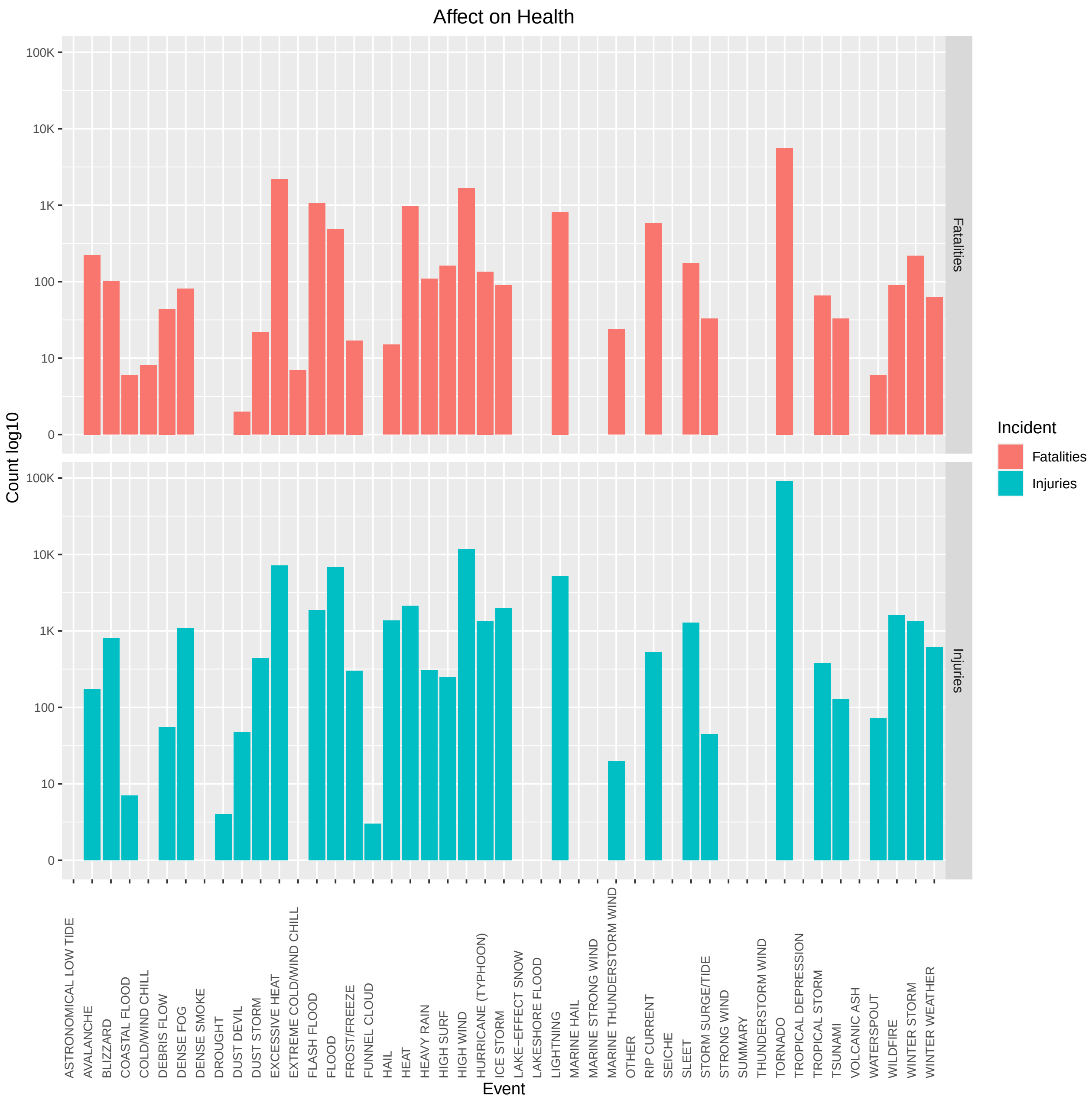

Figure 1: Plot for Fatalities and Injuries caused by Events. The plot gives us an idea of the effect of the events on the health of the community. The y-axis is a pseudo-log scale

```
damage_grp <- gather(damages,PropCrop, Totals, Property:Crops, factor_key=TRUE, na.rm = TRUE)
```

Plotting the damage values to events on a log scale below

```
gg <- ggplot(data = damage_grp, aes(x = EVTYPE_M, y = log10(Totals), fill=PropCrop))+
        geom_col()+
        ggtitle("Damages")+
        xlab("Event") + ylab("Cumulative value log10($)")+
        facet_grid(PropCrop~.)+
        theme(plot.title = element_text(hjust = 0.5),
              axis.text.x = element_text(angle = 90, vjust = 0,
              hjust=0),axis.text=element_text(size=8)) +
        scale_y_continuous(breaks=c(0,3,6,9),
                       labels=c("$0","$1K","$1M","$1B"))

gg
```

Cause of Maximum property damage

```
damages %>% arrange( desc(Property)) %>% filter(Property > 1000000000)
```

```
## # A tibble: 14 x 3
##    EVTYPE_M                  Property       Crops
##    <fct>                        <dbl>       <dbl>
##  1 FLOOD                 150148245680 10734647950
##  2 HURRICANE (TYPHOON)    85356410010  5516117800
##  3 TORNADO                56948858297   414979510
##  4 STORM SURGE/TIDE       47975225500      855000
##  5 HIGH WIND              18903990462  3392365750
##  6 FLASH FLOOD            16967432953  1540690250
##  7 HAIL                   15974041877  3046837650
##  8 WILDFIRE                8496628500   403281630
##  9 TROPICAL STORM          7714440550   694896000
## 10 WINTER STORM            6749997260    32444000
## 11 ICE STORM               3946040310  5022113500
## 12 HEAVY RAIN              3255166192  1132315800
## 13 DROUGHT                 1046106000 13972566000
## 14 SLEET                   1038617257   134663100
```

All damages above $1B, upto 14 events.

Cause for Maximum Crop Damage

Maximum crop damage is caused by:

```
damages %>% arrange( desc(Crops)) %>% filter(Crops > 1000000000)
```

```
## # A tibble: 9 x 3
##   EVTYPE_M                  Property       Crops
##   <fct>                        <dbl>       <dbl>
## 1 DROUGHT                 1046106000 13972566000
## 2 FLOOD                 150148245680 10734647950
## 3 HURRICANE (TYPHOON)    85356410010  5516117800
## 4 ICE STORM               3946040310  5022113500
## 5 HIGH WIND              18903990462  3392365750
## 6 HAIL                   15974041877  3046837650
```

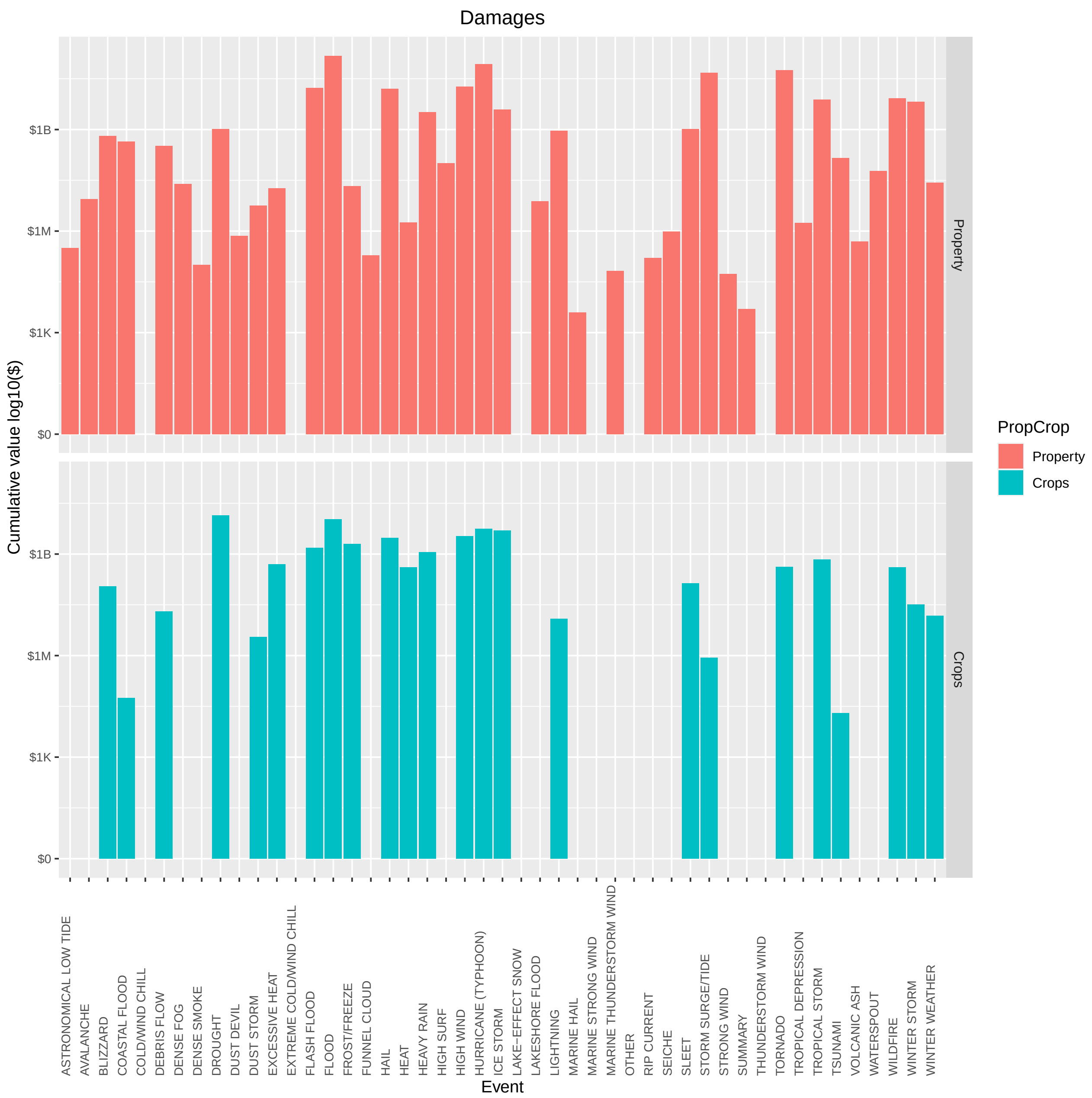

Figure 2: Plot for Property and Crop damages caused by Events. The plot gives us an idea of the damage caused by the events on the community.The y-axis is a pseudo-log scale

```
## 7 FROST/FREEZE          21630700  1997061800
## 8 FLASH FLOOD        16967432953  1540690250
## 9 HEAVY RAIN          3255166192  1132315800
```

All events above caused damages exceeding $1B.

Conclusions

Floods, drought and terrible winters cause most crop damage, while tornadoes, coastal floods, thunderstorms and flash floods cause most property damage. Tornadoes and Hurricanes cause the most fatalities, along with allied phenomena like high wind, lightning and very high temperatures are the second-most dangerous for health of the community. Flash floods are also a major cause for fatalities and injuries.