ReproducibleResearchNOAA

Chau Lui

July 28, 2017

U.S. Weather events and the cost of human lives and properties/crops

This analysis is based on the U.S. National Oceanic and Atmospheric Administration's (NOAA) storm database tracks characteristics of major storms and weather events covering the United States and starting in the year 1950 and ending in November 2011. The focus is on the cost of human lives and the cost of damages.

A couple of questions to address:

1. which types of events caused the most fatalities and injuries. Taking into consideration the toll of each instance (average number of fatalities and injuries) and the total number of fatalities and injuries (sum)
2. which types of events caused the greatest economic damage in terms of properties and crops.

```
knitr::opts_chunk$set(echo = TRUE)

#setwd("C:/WorkingR/Course5Week4")

library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringdist)
library(ggplot2)
```

Data Processing - Load Data

```
# 1. reading in the dataset and/or processing the data

fileUrl <- "https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2"
#

if(!file.exists("repdata-data-StormData.csv.bz2")) download.file(fileUrl, destfile = "repdata-da
ta-StormData.csv.bz2")

stormData <- read.csv("repdata-data-StormData.csv.bz2")

names(stormData)
```

```
##  [1] "STATE__"    "BGN_DATE"   "BGN_TIME"   "TIME_ZONE"  "COUNTY"
##  [6] "COUNTYNAME" "STATE"      "EVTYPE"     "BGN_RANGE"  "BGN_AZI"
## [11] "BGN_LOCATI" "END_DATE"   "END_TIME"   "COUNTY_END" "COUNTYENDN"
## [16] "END_RANGE"  "END_AZI"    "END_LOCATI" "LENGTH"     "WIDTH"
## [21] "F"          "MAG"        "FATALITIES" "INJURIES"   "PROPDMG"
## [26] "PROPDMGEXP" "CROPDMG"    "CROPDMGEXP" "WFO"        "STATEOFFIC"
## [31] "ZONENAMES"  "LATITUDE"   "LONGITUDE"  "LATITUDE_E" "LONGITUDE_"
## [36] "REMARKS"    "REFNUM"
```

Read in the standard Event names from page 6 - 2.1.1 Storm Data Event Table:

NATIONAL WEATHER SERVICE INSTRUCTION 10-1605
(https://d396qusza40orc.cloudfront.net/repdata%2Fpeer2_doc%2Fpd01016005curr.pdf)

From page 6 - 2.1.1 Storm Data Event Table. Copy and paste Storm Data Event Types into a CSV file and read in data.

```
StdEventNames <- read.csv("StormDataEventNames.csv", stringsAsFactors = FALSE)
str(StdEventNames)
```

```
## 'data.frame':    48 obs. of  1 variable:
##  $ EventName: chr  "ASTRONOMICAL LOW TIDE" "AVALANCHE" "BLIZZARD" "COASTAL FLOOD" ...
```

A reusable function to clean up/review and manually adjust EVTYPE typos and restrict categorization.

Closure function ‘cleanUpNames’ accepts a standard list of EVTYPEs (48 items) and a returns a function. Returned function (B) will accept a list (data frame: tmpDF) of unique EVTYPEs for a specific measurement (colName).

1. (B)Function exports a CSV with the EVTYPEs and possible matches to standard list of EVTYPE using ‘amatch()’ function from the “stringdist” package. File name convention is: “ManualMapEVTYPEunique(variable MeasurementColName).csv”
2. User will copy “ManualMapEVTYPEunique(variable MeasurementColName).csv” to “ManualMapEVTYPEunique(variable MeasurementColName)2.csv” and make manual adjustments if

needed.

3. ON SECOND PASS, (B)Function will look for “ManualMapEVTYPEunique(variable MeasurementColName)2.csv” and include ‘adjustedEVTYPE’ column to data frame: tmpDF.
4. (B)Function returns data frame: tmpDF. (See Appendix at end of document)

```
#Usage:

# cleanCol <- cleanUpNames(StdEventNames)
# # 2 columns: "EVTYPE" and focus column
# cleanCol(df, col)

cleanUpNames <- function(stdNames){
      function(tmpDF, colName){
            # tmpDF is dataframe with 2 columns: "EVTYPE" and focus column

            # tmpDF <- stormDataFatal
            # colName = "FATAL"
            # match up EVTYPE to stdNames list

            # Use method = 'jw' and maxDist=10 (trial and error)
             tmpDF <- dplyr::mutate(tmpDF, mapEVTYPEjw10 =
                                    StdEventNames[amatch(toupper(tmpDF[,1]), StdEventNames$Even
tName, method = 'jw', maxDist=10),])
            # Use default (method = 'osa') and maxDist=10
             tmpDF <- dplyr::mutate(tmpDF, mapEVTYPEosa10 =
                                    StdEventNames[amatch(toupper(tmpDF[,1]), StdEventNames$Even
tName, maxDist=10),])
             # If user did not manually adjust, use method = 'jw' and maxDist=10 as default
             tmpDF$adjustedEVTYPE <- tmpDF$mapEVTYPEjw10

            ## write distinct match-ups to .csv to review and adjust manually
            write.csv(tmpDF, file = paste0("ManualMapEVTYPEunique", colName, ".csv"))

            # manual part - user adjust mapping and save file under this name but with a "2" suf
fix
            # on second pass, ManualMapEVTYPEunique2.csv may exist. If so, utilize it.
            if (file.exists(paste0("ManualMapEVTYPEunique", colName, "2.csv"))){
                  # read in file
                  ManualMapEVTYPE <- read.csv(paste0("ManualMapEVTYPEunique", colName, "2.csv"),
 stringsAsFactors = FALSE)

                  # merge tmpDF with adjusted column
                  tmpDF <- merge(x=tmpDF, y=ManualMapEVTYPE[,c("EVTYPE", "adjustedEVTYPE", "Manu
alAdjusted")] )

            }
            #head(tmpDF)
            tmpDF
      }
}
```

Closure function usage:

```
# setup function cleanCol

cleanCol <- cleanUpNames(StdEventNames)
```

Data Processing for cost of human lives analysis

Subset for FATALITIES data: sum, mean, and event type count

```
# Subset for FATALITIES data

# 2 columns: "EVTYPE" and measurement column, either FATALITIES or Injuries, or etc.

# dataframe with 2 columns: "EVTYPE" and measurement column

stormDataFatal <- stormData[(stormData$FATALITIES>0), c("EVTYPE", "FATALITIES")]

# summary(stormDataFatal$FATALITIES)

uniqueMap <- data.frame(EVTYPE = unique(stormDataFatal[,c("EVTYPE")]))

# call function with amatch()/manual adjustments to EVTYPE
uniqueMap <- cleanCol(uniqueMap, "FATAL")


# add "adjustedEVTYPE" from uniqueMap to the initial (non aggregate) data frame

stormDataFatal <- merge(x=stormDataFatal, y=uniqueMap[,c("EVTYPE", "adjustedEVTYPE")] )


# aggregate data based on adjustedEVTYPE
stormDataFatalAgg <- stormDataFatal %>%
      group_by(adjustedEVTYPE) %>%
      summarise(Avg = mean(FATALITIES), Sum = sum(FATALITIES), count=n()) %>%
      mutate(measure = c("FATALITIES"))

#head(stormDataFatalAgg)
```

Subset for INJURIES data: sum, mean, and event type count

```
# Subset for INJURIES data

# 2 columns: "EVTYPE" and measurement column, either FATALITIES or Injuries, or etc.

# dataframe with 2 columns: "EVTYPE" and measurement column

stormDataInjury <- stormData[(stormData$INJURIES>0), c("EVTYPE", "INJURIES")]

uniqueMap <- data.frame(EVTYPE = unique(stormDataInjury[,c("EVTYPE")]))

uniqueMap <- cleanCol(uniqueMap, "Injury")

# add "adjustedEVTYPE" from uniqueMap to the initial (non aggregate) data frame

stormDataInjury <- merge(x=stormDataInjury, y=uniqueMap[,c("EVTYPE", "adjustedEVTYPE")] )


# aggregate data based on adjustedEVTYPE
stormDataInjuryAgg <- stormDataInjury %>%
      group_by(adjustedEVTYPE) %>%
      summarise(Avg = mean(INJURIES), Sum = sum(INJURIES), count=n()) %>%
      mutate(measure = c("INJURIES"))

#head(stormDataInjuryAgg)
```

Sort out top 20 Average number of Fatalities/Injuries per instance of event type

```
(top20FatalAvg <- head(stormDataFatalAgg[order( stormDataFatalAgg$Avg, decreasing = TRUE),],
20))
```

```
## # A tibble: 20 x 5
##             adjustedEVTYPE       Avg   Sum count    measure
##                      <chr>     <dbl> <dbl> <int>      <chr>
##  1                TSUNAMI 16.500000    33     2 FATALITIES
##  2        STORM SURGE/TIDE  6.000000    24     4 FATALITIES
##  3                    HEAT  5.434146  1114   205 FATALITIES
##  4                 TORNADO  3.529632  5658  1603 FATALITIES
##  5          EXCESSIVE HEAT  3.431973  2018   588 FATALITIES
##  6     HURRICANE (TYPHOON)  2.829787   133    47 FATALITIES
##  7          TROPICAL STORM  2.538462    66    26 FATALITIES
##  8              DUST STORM  2.444444    22     9 FATALITIES
##  9                WILDFIRE  2.368421    90    38 FATALITIES
## 10                   SLEET  2.000000     2     1 FATALITIES
## 11              WATERSPOUT  2.000000     6     3 FATALITIES
## 12                BLIZZARD  1.683333   101    60 FATALITIES
## 13            WINTER STORM  1.676923   218   130 FATALITIES
## 14              HEAVY RAIN  1.580645    98    62 FATALITIES
## 15                   FLOOD  1.570957   476   303 FATALITIES
## 16               ICE STORM  1.548387    96    62 FATALITIES
## 17             FLASH FLOOD  1.540785  1020   662 FATALITIES
## 18 EXTREME COLD/WIND CHILL  1.445498   305   211 FATALITIES
## 19               HIGH SURF  1.443038   114    79 FATALITIES
## 20              HEAVY SNOW  1.402174   129    92 FATALITIES
```

```
(top20InjuryAvg <- head(stormDataInjuryAgg[order( stormDataInjuryAgg$Avg, decreasing = TRUE),],
20))
```

```
## # A tibble: 20 x 5
##             adjustedEVTYPE        Avg   Sum count  measure
##                      <chr>      <dbl> <dbl> <int>    <chr>
##  1                TSUNAMI 129.000000   129     1 INJURIES
##  2                    HEAT  44.267857  2479    56 INJURIES
##  3     HURRICANE (TYPHOON)  44.266667  1328    30 INJURIES
##  4                   FLOOD  43.254777  6791   157 INJURIES
##  5          EXCESSIVE HEAT  38.745665  6703   173 INJURIES
##  6               ICE STORM  29.375000  2115    72 INJURIES
##  7                BLIZZARD  16.770833   805    48 INJURIES
##  8          TROPICAL STORM  14.730769   383    26 INJURIES
##  9              WATERSPOUT  12.000000    72     6 INJURIES
## 10                 TORNADO  11.854678 91364  7707 INJURIES
## 11               DENSE FOG  11.032258   342    31 INJURIES
## 12         WINTER WEATHER   10.979592   538    49 INJURIES
## 13             DUST STORM  10.000000   440    44 INJURIES
## 14 EXTREME COLD/WIND CHILL   9.629630   260    27 INJURIES
## 15            WINTER STORM   9.129032  1415   155 INJURIES
## 16              HEAVY SNOW   7.774436  1034   133 INJURIES
## 17         COLD/WIND CHILL   7.500000    60     8 INJURIES
## 18                WILDFIRE   5.098413  1606   315 INJURIES
## 19                    HAIL   4.860714  1361   280 INJURIES
## 20             FLASH FLOOD   4.617571  1787   387 INJURIES
```

Sort out the top 20 Total number of Fatalities/Injuries of each event type

```
(top20FatalSum <- head(stormDataFatalAgg[order( stormDataFatalAgg$Sum, decreasing = TRUE),],
20))
```

```
## # A tibble: 20 x 5
##             adjustedEVTYPE      Avg   Sum count    measure
##                      <chr>    <dbl> <dbl> <int>      <chr>
##  1                TORNADO 3.529632  5658  1603 FATALITIES
##  2          EXCESSIVE HEAT 3.431973  2018   588 FATALITIES
##  3                    HEAT 5.434146  1114   205 FATALITIES
##  4             FLASH FLOOD 1.540785  1020   662 FATALITIES
##  5               LIGHTNING 1.075000   817   760 FATALITIES
##  6             RIP CURRENT 1.131373   577   510 FATALITIES
##  7                   FLOOD 1.570957   476   303 FATALITIES
##  8 EXTREME COLD/WIND CHILL 1.445498   305   211 FATALITIES
##  9               HIGH WIND 1.333333   292   219 FATALITIES
## 10               AVALANCHE 1.285714   225   175 FATALITIES
## 11            WINTER STORM 1.676923   218   130 FATALITIES
## 12       THUNDERSTORM WIND 1.218182   201   165 FATALITIES
## 13         COLD/WIND CHILL 1.256881   137   109 FATALITIES
## 14     HURRICANE (TYPHOON) 2.829787   133    47 FATALITIES
## 15              HEAVY SNOW 1.402174   129    92 FATALITIES
## 16               HIGH SURF 1.443038   114    79 FATALITIES
## 17             STRONG WIND 1.132653   111    98 FATALITIES
## 18                BLIZZARD 1.683333   101    60 FATALITIES
## 19              HEAVY RAIN 1.580645    98    62 FATALITIES
## 20               ICE STORM 1.548387    96    62 FATALITIES
```

```
(top20InjurySum <- head(stormDataInjuryAgg[order( stormDataInjuryAgg$Sum, decreasing = TRUE),],
20))
```

```
## # A tibble: 20 x 5
##         adjustedEVTYPE       Avg   Sum count  measure
##                  <chr>     <dbl> <dbl> <int>    <chr>
##  1             TORNADO 11.854678 91364  7707 INJURIES
##  2               FLOOD 43.254777  6791   157 INJURIES
##  3      EXCESSIVE HEAT 38.745665  6703   173 INJURIES
##  4           LIGHTNING  1.861259  5232  2811 INJURIES
##  5                HEAT 44.267857  2479    56 INJURIES
##  6   THUNDERSTORM WIND  2.594709  2452   945 INJURIES
##  7           ICE STORM 29.375000  2115    72 INJURIES
##  8         FLASH FLOOD  4.617571  1787   387 INJURIES
##  9            WILDFIRE  5.098413  1606   315 INJURIES
## 10           HIGH WIND  2.986056  1499   502 INJURIES
## 11        WINTER STORM  9.129032  1415   155 INJURIES
## 12                HAIL  4.860714  1361   280 INJURIES
## 13 HURRICANE (TYPHOON) 44.266667  1328    30 INJURIES
## 14          HEAVY SNOW  7.774436  1034   133 INJURIES
## 15            BLIZZARD 16.770833   805    48 INJURIES
## 16     WINTER WEATHER  10.979592   538    49 INJURIES
## 17         RIP CURRENT  2.580488   529   205 INJURIES
## 18          DUST STORM 10.000000   440    44 INJURIES
## 19      TROPICAL STORM 14.730769   383    26 INJURIES
## 20           DENSE FOG 11.032258   342    31 INJURIES
```

Find which event types (adjustedEVTYPE) are on both of the top 20 Avg and Sum lists

```
# Find the event types that are in both lists of highest number of total injuries (sum) and most
 injuries per event (average)
InjuryBothTop20SumAvg <- intersect(top20InjuryAvg$adjustedEVTYPE,top20InjurySum$adjustedEVTYPE)

# Find the event types that are in both lists of highest number of total fatalities (sum) and mo
st fatalities per event (average)
FatalBothTop20SumAvg <- intersect(top20FatalAvg$adjustedEVTYPE,top20FatalSum$adjustedEVTYPE)

# Find the event types that are in both lists of highest number of total injuries and fatalities
both <- intersect(InjuryBothTop20SumAvg, FatalBothTop20SumAvg)

# Subset on these event types
(stormDataFatalAgg[(stormDataFatalAgg$adjustedEVTYPE %in% both), ])
```

```
## # A tibble: 10 x 5
##         adjustedEVTYPE      Avg   Sum count    measure
##                  <chr>    <dbl> <dbl> <int>      <chr>
##  1            BLIZZARD 1.683333   101    60 FATALITIES
##  2      EXCESSIVE HEAT 3.431973  2018   588 FATALITIES
##  3         FLASH FLOOD 1.540785  1020   662 FATALITIES
##  4               FLOOD 1.570957   476   303 FATALITIES
##  5                HEAT 5.434146  1114   205 FATALITIES
##  6          HEAVY SNOW 1.402174   129    92 FATALITIES
##  7 HURRICANE (TYPHOON) 2.829787   133    47 FATALITIES
##  8           ICE STORM 1.548387    96    62 FATALITIES
##  9             TORNADO 3.529632  5658  1603 FATALITIES
## 10        WINTER STORM 1.676923   218   130 FATALITIES
```

```
(stormDataInjuryAgg[(stormDataInjuryAgg$adjustedEVTYPE %in% both), ])
```

```
## # A tibble: 10 x 5
##         adjustedEVTYPE       Avg   Sum count  measure
##                  <chr>     <dbl> <dbl> <int>    <chr>
##  1            BLIZZARD 16.770833   805    48 INJURIES
##  2      EXCESSIVE HEAT 38.745665  6703   173 INJURIES
##  3         FLASH FLOOD  4.617571  1787   387 INJURIES
##  4               FLOOD 43.254777  6791   157 INJURIES
##  5                HEAT 44.267857  2479    56 INJURIES
##  6          HEAVY SNOW  7.774436  1034   133 INJURIES
##  7 HURRICANE (TYPHOON) 44.266667  1328    30 INJURIES
##  8           ICE STORM 29.375000  2115    72 INJURIES
##  9             TORNADO 11.854678 91364  7707 INJURIES
## 10        WINTER STORM  9.129032  1415   155 INJURIES
```

```
harmHealthAgg <- rbind(stormDataFatalAgg[(stormDataFatalAgg$adjustedEVTYPE %in% both), ], stormD
ataInjuryAgg[(stormDataInjuryAgg$adjustedEVTYPE %in% both), ])
```

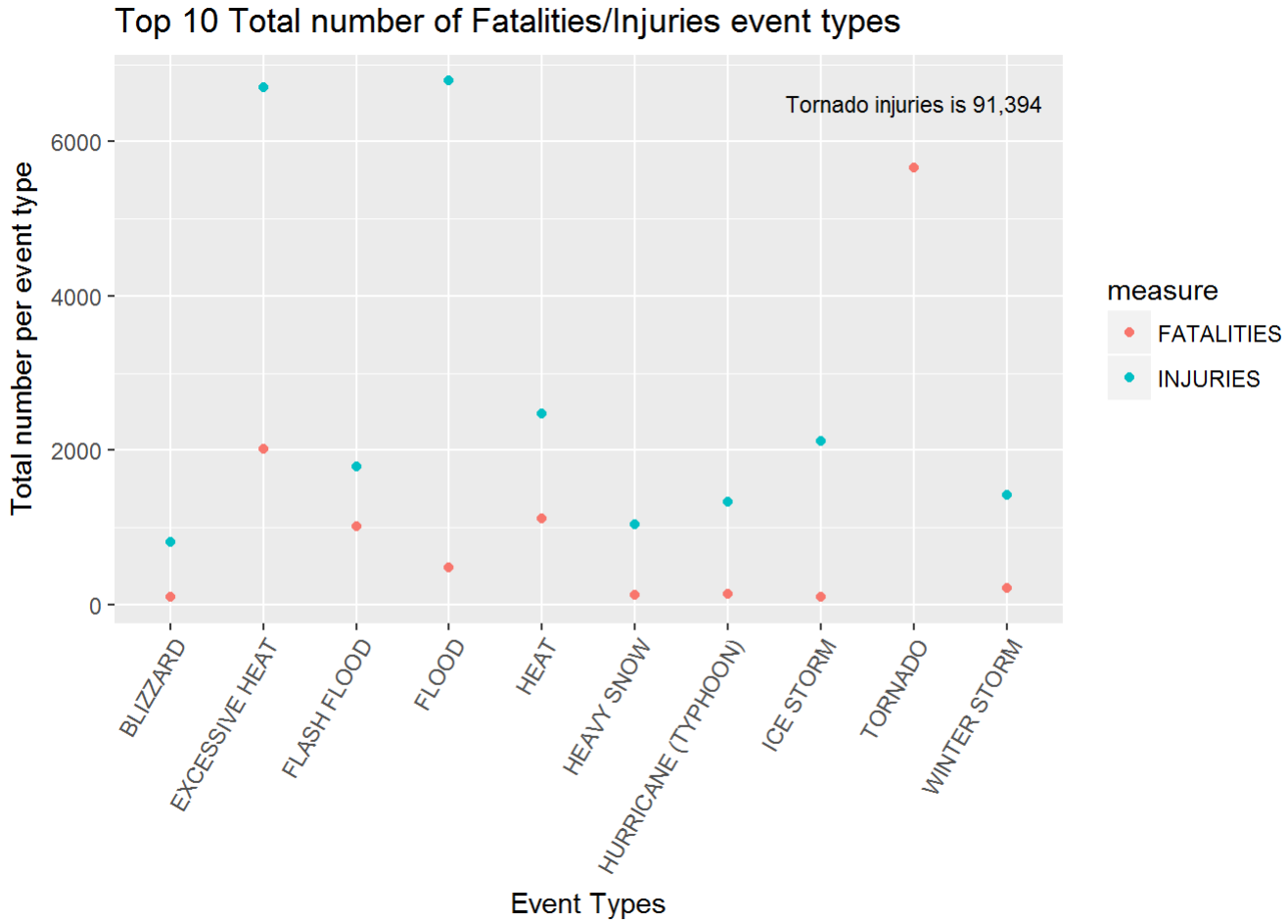
Plot out the total number of Fatalities/Injuries for event types

The number of injuries due to TORNADO event (91,394) is an outlier compared to the next highest count from FLOOD (6,791). Change the y-axis plotting range to exclude the outlier.

```
(rngSum <- range(harmHealthAgg[harmHealthAgg$Sum < max(harmHealthAgg$Sum),c("Sum")]))
```

```
## [1]   96 6791
```

```
q <- qplot(adjustedEVTYPE, Sum, data=harmHealthAgg, color = measure )
q + theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
      coord_cartesian(ylim=rngSum) +
      ggtitle("Top 10 Total number of Fatalities/Injuries event types") +
      labs(x="Event Types",y="Total number per event type") +
      annotate("text", label = "Tornado injuries is 91,394", x = 9, y = 6500, size = 3, colour =
 "black")
```

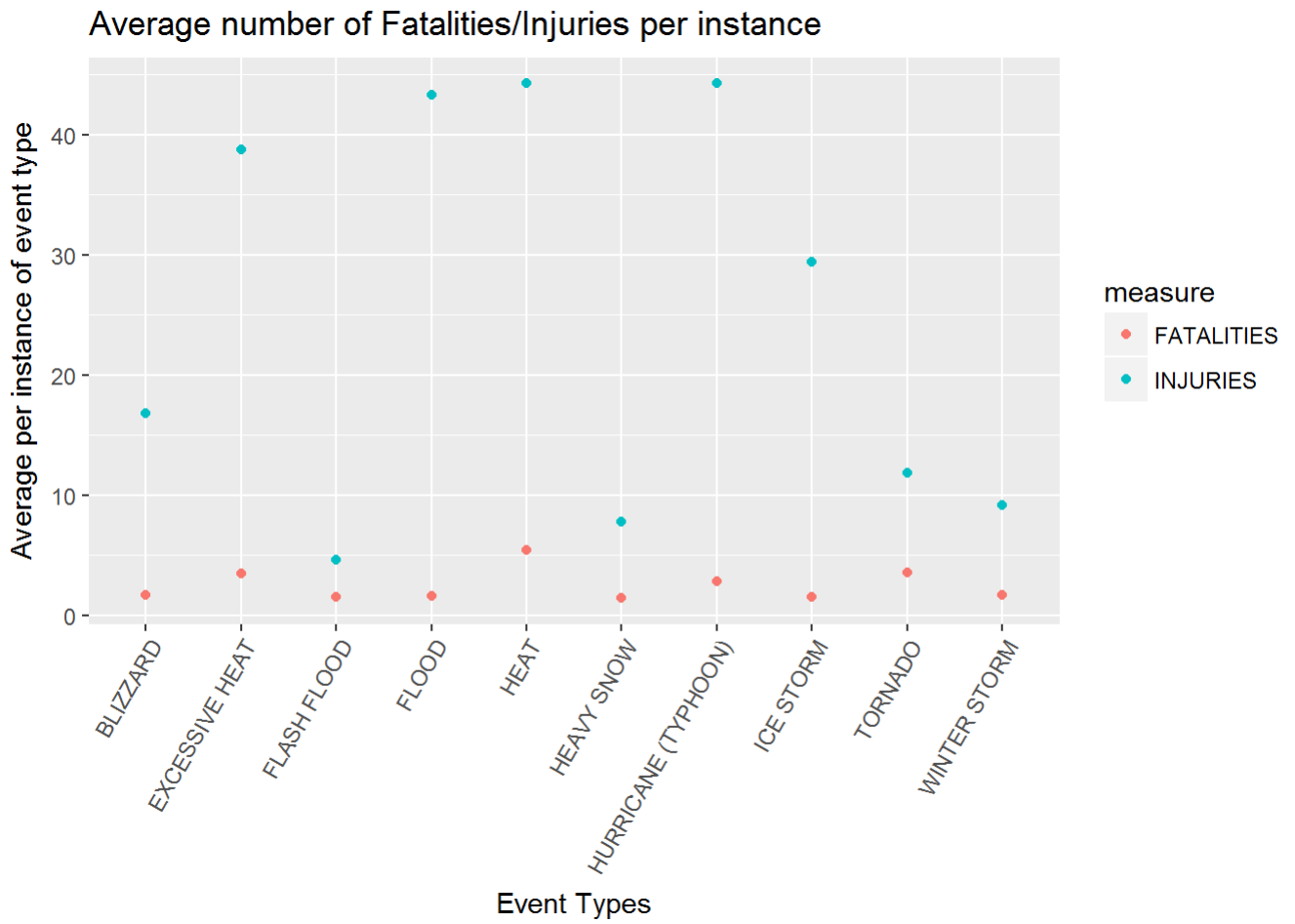


Plot out the average number of Fatalities/Injuries for event types

```
(rngAvg <- range(harmHealthAgg[, c("Avg")]))
```

```
## [1]  1.402174 44.267857
```

```
q <- qplot(adjustedEVTYPE, Avg, data=harmHealthAgg, color = measure )
q + theme(axis.text.x = element_text(angle = 60, hjust = 1)) +
      coord_cartesian(ylim=rngAvg) +
      ggtitle("Average number of Fatalities/Injuries per instance") +
      labs(x="Event Types",y="Average per instance of event type")
```



Results

Event types Tornado, Flash Flood, and Heat top list for cost to human lives.

Based on the above lists to represent the event types that caused the most harm to human lives, tornado tops the list of total number of fatalities (5,658) and injuries (91,394) compared to the next highest count of injuries (6,791) from flood. Heat/excessive heat is second on the list for fatalities, totaling more than 3,000 and injuries in 9,000 plus.

On average, tsumanis/Storm surge/tide cause the most fatalities and injuries per instance but occur infrequently. Heat/excessive heat, on average, caused the most fatalities and injuries and occur frequently.

Lightning and thunderstorm wind round up the list with high number of injuries and frequent occurrence.

Data Processing for cost of property/crop damages analysis

PROPDMGEXP and CROPDMGEXP columns of the database

https://rstudio-pubs-static.s3.amazonaws.com/58957_37b6723ee52b455990e149edde45e5b6.html (https://rstudio-pubs-static.s3.amazonaws.com/58957_37b6723ee52b455990e149edde45e5b6.html)

Based on the site on how to handle exponent value of PROPDMGEXP and CROPDMGEXP columns of the database, create a conversion mapping.

These are possible values of CROPDMGEXP and PROPDMGEXP:

- H,h,K,k,M,m,B,b,+,-,?,0,1,2,3,4,5,6,7,8, and blank-character
- H,h = hundreds = 100
- K,k = kilos = thousands = 1,000
- M,m = millions = 1,000,000
- B,b = billions = 1,000,000,000
- (+) = 1
- (-) = 0
- (?) = 0
- blank/empty character = 0
- numeric 0..8 = 10

```
stormDataPropDamage <- stormData[(stormData$PROPDMG>0), c("EVTYPE", "PROPDMG", "PROPDMGEXP")]

DMGEXPList <-c("H","h","K","k","M","m","B","b","+","-","?","0","1","2","3","4","5","6","7","8",
" ")

conv <- c(100, 100, 1000, 1000, 100000, 100000,
1000000000,1000000000,1,0,0,10,10,10,10,10,10,10,10,10,0)

EXPMap <- data.frame(DMGEXPList=DMGEXPList, conv =conv)


# Include 'conv' column from EXPMap for each PROPDMGEXP
stormDataPropDamage <- merge(x=stormDataPropDamage, y=EXPMap, by.x="PROPDMGEXP", by.y="DMGEXPLis
t")

#
# stormDataPropDamage <- mutate(stormDataPropDamage, PROPDMGValue = PROPDMG * conv)
#
#  sum(is.na(stormDataPropDamage$conv))
#
#  sum(is.na(stormDataPropDamage$PROPDMG))
#  sum(is.na(stormDataPropDamage$PROPDMGValue))
#
#  PROPDMGValue
#  unique(stormDataPropDamage$conv)

# do EVTYPE clean-up
uniqueMap <- data.frame(EVTYPE = unique(stormDataPropDamage[,c("EVTYPE")]))

uniqueMap <- cleanCol(uniqueMap, "PROPDMG")


# merge column "adjustedEVTYPE" from uniqueMap
stormDataPropDamage <- merge(stormDataPropDamage, uniqueMap[, c("EVTYPE", "adjustedEVTYPE")])

stormDataPropDamageGrp <- stormDataPropDamage %>%
      mutate(PropDmgValue = PROPDMG * conv) %>%
      group_by(adjustedEVTYPE) %>%
      summarise(Avg = mean(PropDmgValue), Sum = sum(PropDmgValue), count=n()) %>%
      arrange(desc(Sum))

#summary(stormDataPropDamageGrp)
```

Show top 5 property damage values in M (millions) on plot

```
(Top5PropDmg <-head(stormDataPropDamageGrp, 5))
```

```
## # A tibble: 5 x 4
##        adjustedEVTYPE         Avg          Sum count
##                 <chr>       <dbl>        <dbl> <int>
## 1               FLOOD  12373358.6 125577216554 10149
## 2 HURRICANE (TYPHOON) 362248192.4  75347624010   208
## 3    STORM SURGE/TIDE 212376100.0  46722742000   220
## 4             TORNADO    381858.3  14913474647 39055
## 5         FLASH FLOOD    423686.6   8894876631 20994
```

```
Top5PropDmg$PropDmgValM <- Top5PropDmg$Sum / 1000000
Top5PropDmg$PropDmgAvgM <- Top5PropDmg$Avg / 1000000
```

Do the same for Crop Damage

```
stormDataCropDamage <- stormData[(stormData$CROPDMG>0), c("EVTYPE", "CROPDMG", "CROPDMGEXP")]

stormDataCropDamage <- merge(x=stormDataCropDamage, y=EXPMap, by.x="CROPDMGEXP", by.y="DMGEXPLis
t")

uniqueMap <- data.frame(EVTYPE = unique(stormDataCropDamage[,c("EVTYPE")]))

uniqueMap <- cleanCol(uniqueMap, "CROPDMG")


# merge column "adjustedEVTYPE"
stormDataCropDamage <- merge(stormDataCropDamage, uniqueMap[, c("EVTYPE", "adjustedEVTYPE")])

stormDataCropDamageGrp <- stormDataCropDamage %>%
      mutate(CropDmgValue = CROPDMG * conv) %>%
           group_by(adjustedEVTYPE) %>%
      summarise(Avg = mean(CropDmgValue), Sum = sum(CropDmgValue), count=n()) %>%
      arrange(desc(Sum))


#summary(stormDataPropDamageGrp)
```

Show top 5 Crop Damage values in M (millions) on plot

```
(Top5CropDmg <- head(stormDataCropDamageGrp,5))
```

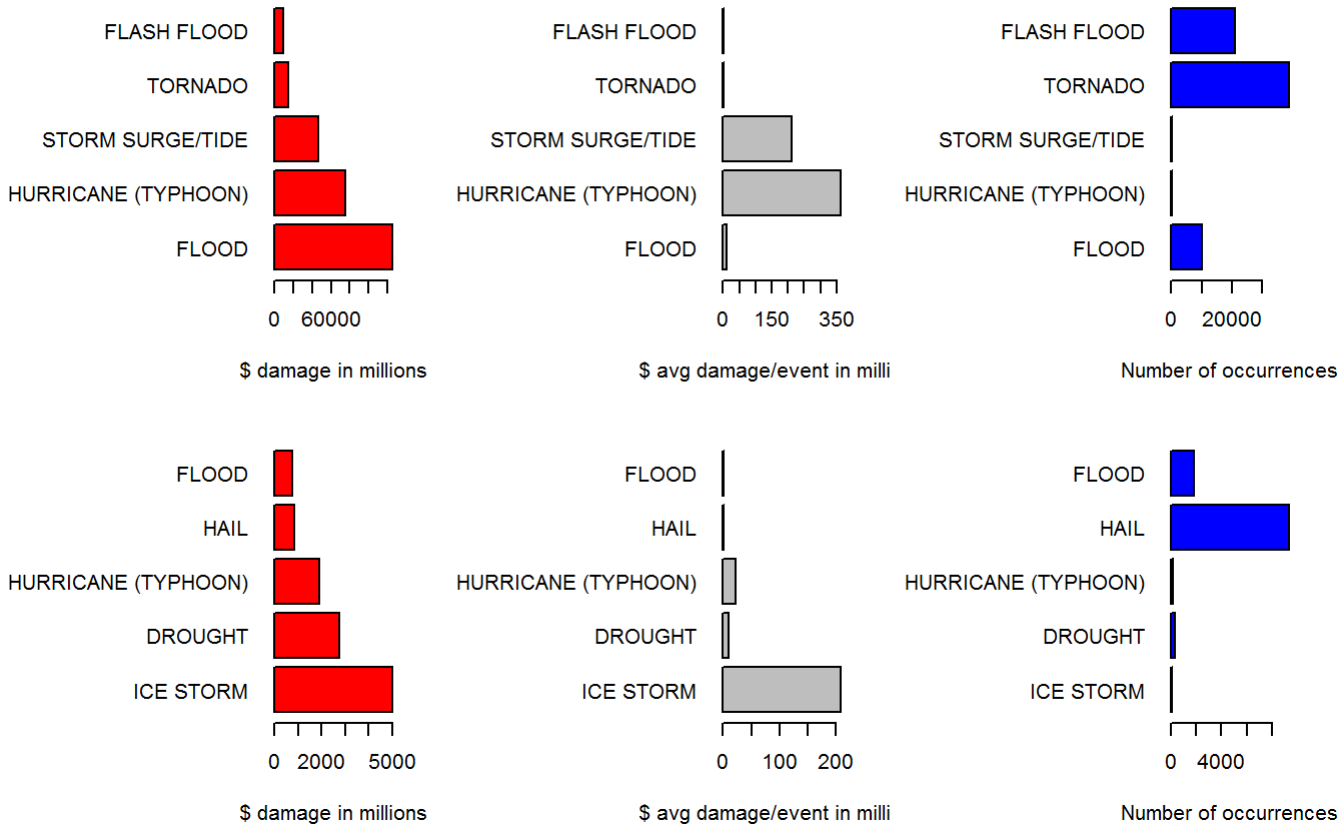
```
## # A tibble: 5 x 4
##        adjustedEVTYPE         Avg        Sum count
##                 <chr>       <dbl>      <dbl> <int>
## 1           ICE STORM 208487854.2 5003708500    24
## 2             DROUGHT  10640629.9 2766563780   260
## 3 HURRICANE (TYPHOON)  21524278.7 1915660800    89
## 4                HAIL     87621.0  822147800  9383
## 5               FLOOD    395912.7  728083450  1839
```

```
Top5CropDmg$CropDmgValM <- Top5CropDmg$Sum / 1000000
Top5CropDmg$CropDmgAvgM <- Top5CropDmg$Avg / 1000000
```

Plot the total amount of damage (in millions), averge damage per event (in millions), and the number of occurrences of top 5 events in terms of total amount of damage

```
#set up plot line color
plotCol <- c("red","gray", "blue")

par(mfrow = c(2, 3), oma = c(0,0,3,0), mar = c(4.3, 11, 2, 2), las=1 )
# par(mfrow = c(2, 3), oma = c(0,0,3,0), mar = c(6, 12, 4, 3), las=1 )

with(Top5PropDmg, barplot(PropDmgValM, col=plotCol[1], horiz=TRUE, names.arg = adjustedEVTYPE, x
lab = "$ damage in millions"))
with(Top5PropDmg, barplot(PropDmgAvgM, col=plotCol[2], horiz=TRUE, names.arg = adjustedEVTYPE, x
lab = "$ avg damage/event in millions"))
with(Top5PropDmg, barplot(count, col=plotCol[3], horiz=TRUE, names.arg = adjustedEVTYPE, xlab =
"Number of occurrences"))

title(main = "Properties (top) and Crops (bottom) damage due to weather events from 1950 to Nove
mber 2011", outer = TRUE)

with(Top5CropDmg, barplot(CropDmgValM, col=plotCol[1], horiz=TRUE, names.arg = adjustedEVTYPE, x
lab = "$ damage in millions"))
#title(sub="Crops Damages")

with(Top5CropDmg, barplot(CropDmgAvgM, col=plotCol[2], horiz=TRUE, names.arg = adjustedEVTYPE,
xlab = "$ avg damage/event in millions"))

with(Top5CropDmg, barplot(count, col=plotCol[3], horiz=TRUE, names.arg = adjustedEVTYPE, xlab =
"Number of occurrences"))
```

Properties (top) and Crops (bottom) damage due to weather events from 1950 to November 2011



Results

Event types Ice Storm and Drought top list for cost for crops damages while Flood and Hurricane (Typhoon) top for property damages.

Ice Storm and Drought do not occur frequently but they caused the majority of crops damages.

Most property damages appeared to be water related with event types: Flood, Flash Flood, Hurricane (Typhoon), and Storm Surge/Tide.

Appendix:

Additional working files for reproducible factor, manually adjusted EVTYPE:

Fatalities EVTYPE manual adjustments
(https://github.com/ChauLui/ReproducibleResearch/blob/master/ManualMapEVTYPEuniqueFATAL2.csv)

Injuries EVTYPE manual adjustments
(https://github.com/ChauLui/ReproducibleResearch/blob/master/ManualMapEVTYPEuniqueInjury2.csv)

Crop Damage EVTYPE manual adjustments
(https://github.com/ChauLui/ReproducibleResearch/blob/master/ManualMapEVTYPEuniqueCROPDMG2.csv)

Property Damage EVTYPE manual adjustments
(https://github.com/ChauLui/ReproducibleResearch/blob/master/ManualMapEVTYPEuniquePROPDMG2.csv)