# Economic and Human Impact of Storms and other Weather Events in the United States

### 1. Synopsis

This assignment uses NOAA (National Oceanic & Atmospheric Administration) Storm Database for analysis purpose. The source used for this analysis was picked from here - https://d396qusza40orc.cloudfront.net/repdata%2Fdata%2FStormData.csv.bz2 and analysis period is between 1950 and November 2011. The database contains characteristics of Major weather events in the United States of America. This assignment fetches that data, cleans it up, organizes it, and then aggregate the following - 1. Most harmful events to human population health. 2. Events that have greatest economic consequences.

The conclusion is that the Tornadoes & Excessive Heat and Tornadoes & Thunderstorm Winds are the most harmful events to human population health from Fatalities and Injuries perspective respectively. Similarly, Floods & Hurricane - Typhoons, Droughts & Floods and Floods and Hurricane - Typhoons are the events having greatest economic consequences from Property Damages, Crop Damages and Overall Damages perspective respectively. It is also concluded that events harmful to human health are not at all correlated to events with economic consequences.

R version used for analysis is "R version 3.3.1 (2016-06-21)". OS used is "x86_64, Linux-gnu".

### 2. Basic settings

```
echo = TRUE  # Make code always visible
options(scipen = 1)  # Turn off scientific notations for numbers
```

### 3. Loading Libraries

Load all the needed libraries for data extraction, caching, processing data, and plotting graphs

```
library(R.cache)
```

```
## R.cache v0.12.0 (2015-11-12) successfully loaded. See ?R.cache for help.
```

```
library(R.utils)
```

```
## Loading required package: R.oo
```

```
## Loading required package: R.methodsS3
```

```
## R.methodsS3 v1.7.1 (2016-02-15) successfully loaded. See ?R.methodsS3 for help.
```

```
## R.oo v1.20.0 (2016-02-17) successfully loaded. See ?R.oo for help.
```

```
##
## Attaching package: 'R.oo'
```

```
## The following objects are masked from 'package:methods':
##
##     getClasses, getMethods

## The following objects are masked from 'package:base':
##
##     attach, detach, gc, load, save

## R.utils v2.3.0 (2016-04-13) successfully loaded. See ?R.utils for help.

##
## Attaching package: 'R.utils'

## The following object is masked from 'package:utils':
##
##     timestamp

## The following objects are masked from 'package:base':
##
##     cat, commandArgs, getOption, inherits, isOpen, parse, warnings
```

```
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(ggplot2)
```

### 4. Downloading and extracting files

```
# 4.1. Check if data directory exists or create it
if(!file.exists("./data"))
{
      dir.create("./data")
}

# 4.2. Check if the zip file exists or download it
if(!file.exists("./data/StormData.csv.bz2"))
{
    fileUrl = "https://d396qusza40orc.cloudfront.net/repdata/data/StormData.csv.bz2"
    destPath = "./data/StormData.csv.bz2"
```

```
    download.file(fileUrl, destPath)
}

# 4.3. Check if CSV file exists or extract it from zip
if(!file.exists("./data/StormData.csv"))
{
    filePath = "./data/StormData.csv.bz2"
    destPath = "./data/StormData.csv"
    bunzip2(filePath, destPath, remove=FALSE)
}
```

### 5. Data Processing

```
# 5.1. Load the CSV files as raw data
rawStormData = read.csv("./data/StormData.csv", header = TRUE, sep = ",")

# 5.2. Verify the structure and dimensions of raw data frame
str(rawStormData)
```

```
## 'data.frame':    902297 obs. of  37 variables:
##  $ STATE__    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ BGN_DATE   : Factor w/ 16335 levels "10/10/1954 0:00:00",..: 6523 6523 4213 11116 1426 1426 1462 28
##  $ BGN_TIME   : Factor w/ 3608 levels "000","0000","00:00:00 AM",..: 212 257 2645 1563 2524 3126 122 1
##  $ TIME_ZONE  : Factor w/ 22 levels "ADT","AKS","AST",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ COUNTY     : num  97 3 57 89 43 77 9 123 125 57 ...
##  $ COUNTYNAME: Factor w/ 29601 levels "","5NM E OF MACKINAC BRIDGE TO PRESQUE ISLE LT MI",..: 13513 1
##  $ STATE      : Factor w/ 72 levels "AK","AL","AM",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ EVTYPE     : Factor w/ 985 levels "?","ABNORMALLY DRY",..: 830 830 830 830 830 830 830 830 830 830
##  $ BGN_RANGE  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ BGN_AZI    : Factor w/ 35 levels "","E","Eas","EE",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ BGN_LOCATI: Factor w/ 54429 levels "","?","(01R)AFB GNRY RNG AL",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ END_DATE   : Factor w/ 6663 levels "","10/10/1993 0:00:00",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ END_TIME   : Factor w/ 3647 levels "","?","0000",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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    : Factor w/ 24 levels "","E","ENE","ESE",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ END_LOCATI: Factor w/ 34506 levels "","(0E4)PAYSON ARPT",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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: Factor w/ 19 levels "","-","?","+",..: 17 17 17 17 17 17 17 17 17 17 ...
##  $ CROPDMG    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ CROPDMGEXP: Factor w/ 9 levels "","?","0","2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ WFO        : Factor w/ 542 levels "","2","43","9V9",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ STATEOFFIC: Factor w/ 250 levels "","ALABAMA, Central",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ ZONENAMES  : Factor w/ 25112 levels "","
```

```
##  $ 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   : Factor w/ 436781 levels "","  ","  ","   ",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ REFNUM    : num  1 2 3 4 5 6 7 8 9 10 ...
```

```r
head(rawStormData)
```

```
##   STATE__           BGN_DATE BGN_TIME TIME_ZONE COUNTY COUNTYNAME STATE
## 1       1  4/18/1950 0:00:00     0130       CST     97     MOBILE    AL
## 2       1  4/18/1950 0:00:00     0145       CST      3    BALDWIN    AL
## 3       1  2/20/1951 0:00:00     1600       CST     57    FAYETTE    AL
## 4       1   6/8/1951 0:00:00     0900       CST     89    MADISON    AL
## 5       1 11/15/1951 0:00:00     1500       CST     43    CULLMAN    AL
## 6       1 11/15/1951 0:00:00     2000       CST     77 LAUDERDALE    AL
##    EVTYPE BGN_RANGE BGN_AZI BGN_LOCATI END_DATE END_TIME COUNTY_END
## 1 TORNADO         0                                               0
## 2 TORNADO         0                                               0
## 3 TORNADO         0                                               0
## 4 TORNADO         0                                               0
## 5 TORNADO         0                                               0
## 6 TORNADO         0                                               0
##   COUNTYENDN END_RANGE END_AZI END_LOCATI LENGTH WIDTH F MAG FATALITIES
## 1         NA         0                    14.0   100 3   0          0
## 2         NA         0                     2.0   150 2   0          0
## 3         NA         0                     0.1   123 2   0          0
## 4         NA         0                     0.0   100 2   0          0
## 5         NA         0                     0.0   150 2   0          0
## 6         NA         0                     1.5   177 2   0          0
##   INJURIES PROPDMG PROPDMGEXP CROPDMG CROPDMGEXP WFO STATEOFFIC ZONENAMES
## 1       15    25.0          K       0
## 2        0     2.5          K       0
## 3        2    25.0          K       0
## 4        2     2.5          K       0
## 5        2     2.5          K       0
## 6        6     2.5          K       0
##   LATITUDE LONGITUDE LATITUDE_E LONGITUDE_ REMARKS REFNUM
## 1     3040      8812       3051       8806              1
## 2     3042      8755          0          0              2
## 3     3340      8742          0          0              3
## 4     3458      8626          0          0              4
## 5     3412      8642          0          0              5
## 6     3450      8748          0          0              6
```

The columns required for further analysis and answering our queries are: BGN_DATE Beginning Date (to decide the threshold or starting year) EVTYPE Event Type FATALITIES Number of human fatalities due to event INJURIES Number of human injuries due to event PROPDMG Property Damage in USD PROPDMGEXP Magnitude of Property Damage (Thousands, Millions) USDs CROPDMG Crop Damage in USD CROPDMGEXP Magnitude of Crop Damage (Thousands, Millions) USDs

```r
# 5.3. Remove the unwanted columns from raw data frame
columnsRequired = c("BGN_DATE", "EVTYPE", "FATALITIES", "INJURIES", "PROPDMG", "PROPDMGEXP", "CROPDMG",
```

```
modStormData = rawStormData[, columnsRequired]

# 5.4. Verify if there is NA in data
sum(is.na(modStormData))
```

```
## [1] 0
```

```
# 5.5. Convert the EVTTYPE, PROPDMGEXP and CROPDMGEXP data into upper case for further usage
modStormData$EVTYPE=toupper(modStormData$EVTYPE)
modStormData$PROPDMGEXP = toupper(modStormData$PROPDMGEXP)
modStormData$CROPDMGEXP = toupper(modStormData$CROPDMGEXP)
```

By looking at the event types, it is understood that the Event Type data has to be cleaned up for better results and understanding, e.g.: "TSTM WIND", "TSTM WIND (G40)"" and "MARINE TSTM WIND" These are the events same as "MARINE THUNDERSTORM WIND" but with different descriptions. Such event types are documented in the page 6 of the storm data documentation (https://d396qusza40orc.cloudfront.net/repdata%2Fpeer2_doc%2Fpd01016005curr.pdf), so it needs to clean the dataset and combine the same types of events following the table on page 6 of the documentation.

```
# 5.6. Factor the EVTTYPE to avoid duplicates and arrive at the correct measures
modStormData[modStormData$EVTYPE == "HURRICANE/TYPHOON", "EVTYPE"] = "HURRICANE-TYPHOON"
modStormData[modStormData$EVTYPE == "HURRICANE", "EVTYPE"] = "HURRICANE-TYPHOON"
modStormData[modStormData$EVTYPE == "RIVER FLOOD", "EVTYPE"] = "FLOOD"
modStormData[modStormData$EVTYPE == "THUNDERSTORM WINDS", "EVTYPE"] = "THUNDERSTORM WIND"
modStormData[modStormData$EVTYPE == "TSTM WIND", "EVTYPE"] = "THUNDERSTORM WIND"
modStormData[modStormData$EVTYPE == "URBAN/SML STREAM FLD", "EVTYPE"] = "HEAVY RAIN"
modStormData[modStormData$EVTYPE == "MARINE TSTM WIND", "EVTYPE"] = "MARINE THUNDERSTORM WIND"
modStormData[modStormData$EVTYPE == "WILD/FOREST FIRE", "EVTYPE"] = "WILDFIRE"
modStormData[modStormData$EVTYPE == "marinethunderstormwind/hail", "EVTYPE"] = "MARINE THUNDERSTORM WIN
modStormData[modStormData$EVTYPE == "TSTM WIND/HAIL", "EVTYPE"] = "MARINE THUNDERSTORM WIND"
modStormData[modStormData$EVTYPE == "flashflooding", "EVTYPE"] = "FLASH FLOOD"
modStormData[modStormData$EVTYPE == "FLOOD/FLASH FLOOD", "EVTYPE"] = "FLASH FLOOD"
modStormData[modStormData$EVTYPE == "WINTER data/MIX", "EVTYPE"] = "WINTER data"
modStormData[modStormData$EVTYPE == "RIP CURRENTS", "EVTYPE"] = "RIP CURRENT"
modStormData[modStormData$EVTYPE == "DENSEDENSEFOG", "EVTYPE"] = "DENSE FOG"
modStormData[modStormData$EVTYPE == "STRONG WINDS", "EVTYPE"] = "ASTROMICAL LOW TIDE"
modStormData[modStormData$EVTYPE == "COASTAL FLOODING", "EVTYPE"] = "COASTAL FLOOD"
modStormData[modStormData$EVTYPE == "RECORD WARMTH", "EVTYPE"] = "HEAT"
modStormData[modStormData$EVTYPE == "RECORD HEAT", "EVTYPE"] = "HEAT"
modStormData[modStormData$EVTYPE == "FREEZE", "EVTYPE"] = "FROST/FREEZE"
modStormData[modStormData$EVTYPE == "HEATWAVE", "EVTYPE"] = "EXCESSIVE HEAT"
```

Finalize the Cut Off Year (if not 1950) based on the details provided in Assignment Statement ("The events in the database start in the year 1950 and end in November 2011. In the earlier years of the database there are generally fewer events recorded, most likely due to a lack of good records. More recent years should be considered more complete.)".

```
# 5.7. Find the Cut Off Year to include the most recent data
totalNumberOfObservations = nrow(modStormData)
cutOffPercentage = 0.75
cutOffObservations = round(totalNumberOfObservations * cutOffPercentage)
```

```r
# 5.7.1. add columns for date calculations based on BGN_DATE
modStormData$year = as.numeric(format(as.Date(modStormData$BGN_DATE, format = "%m/%d/%Y"), "%Y"))

# 5.7.2. create dataset with count per year, reverse the recordset, create running total
yearRecords = as.data.frame(table(modStormData$year))
yearRecords[order(yearRecords$Var1, decreasing = TRUE), ]
```

```
##    Var1  Freq
## 62 2011 62174
## 61 2010 48161
## 60 2009 45817
## 59 2008 55663
## 58 2007 43289
## 57 2006 44034
## 56 2005 39184
## 55 2004 39363
## 54 2003 39752
## 53 2002 36293
## 52 2001 34962
## 51 2000 34471
## 50 1999 31289
## 49 1998 38128
## 48 1997 28680
## 47 1996 32270
## 46 1995 27970
## 45 1994 20631
## 44 1993 12607
## 43 1992 13534
## 42 1991 12522
## 41 1990 10946
## 40 1989 10410
## 39 1988  7257
## 38 1987  7367
## 37 1986  8726
## 36 1985  7979
## 35 1984  7335
## 34 1983  8322
## 33 1982  7132
## 32 1981  4517
## 31 1980  6146
## 30 1979  4279
## 29 1978  3657
## 28 1977  3728
## 27 1976  3768
## 26 1975  4975
## 25 1974  5386
## 24 1973  4463
## 23 1972  2168
## 22 1971  3471
## 21 1970  3215
## 20 1969  2926
## 19 1968  3312
## 18 1967  2688
```

```
## 17 1966  2388
## 16 1965  2855
## 15 1964  2348
## 14 1963  1968
## 13 1962  2389
## 12 1961  2246
## 11 1960  1945
## 10 1959  1813
## 9  1958  2213
## 8  1957  2184
## 7  1956  1703
## 6  1955  1413
## 5  1954   609
## 4  1953   492
## 3  1952   272
## 2  1951   269
## 1  1950   223
```

```
yearRecords$runningTotal = cumsum(yearRecords$Freq)
yearRecords$Var1=as.numeric(as.character(yearRecords$Var1))
cutOffYear = min(yearRecords[yearRecords$runningTotal < cutOffObservations, 1])

modStormData = modStormData[modStormData$year >= cutOffYear, ]
endYear = max(modStormData$year)
```

### 6. Results

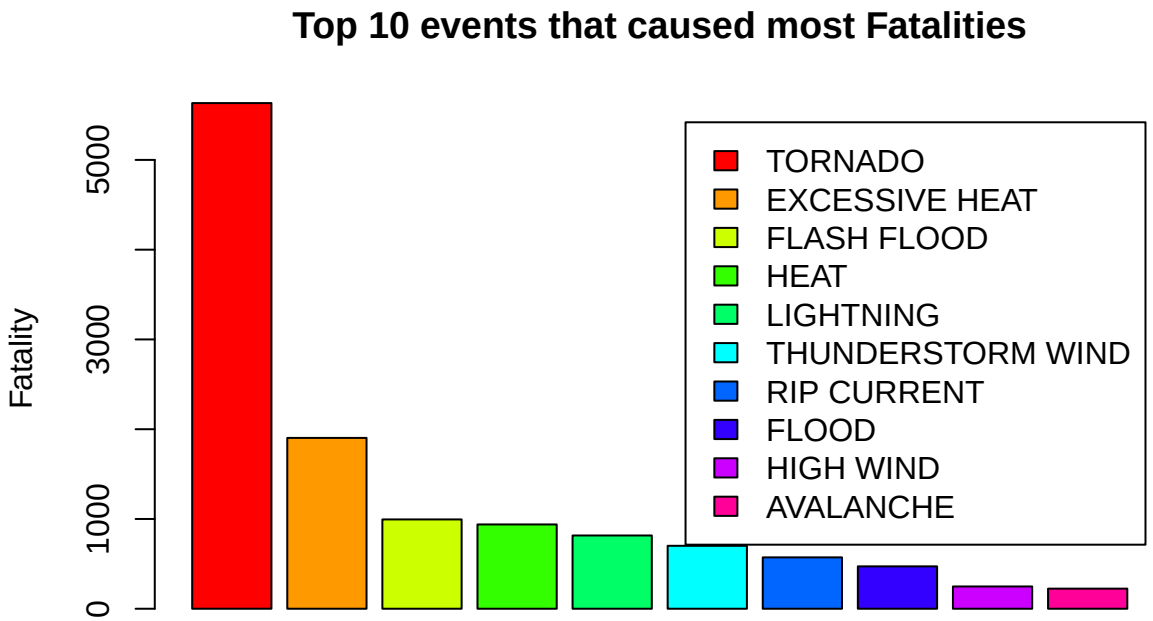
```
# 6.1. Across the United States, which types of events (as indicated in the EVTYPE variable) are most h

# 6.1.1. finding the worst injuries
injStats = aggregate(INJURIES~EVTYPE, data = modStormData, sum)
# exclude the results with zero injuries
injStats = injStats [injStats$INJURIES>0,]
# arrange the results in descending order to find worst 10 results
injDescr = injStats [order(injStats$INJURIES, decreasing = TRUE),]
head (injDescr, n=10)
```

```
##                EVTYPE INJURIES
## 744           TORNADO    91346
## 675 THUNDERSTORM WIND     9353
## 148             FLOOD     6791
## 109    EXCESSIVE HEAT     6525
## 405         LIGHTNING     5230
## 235              HEAT     2150
## 380         ICE STORM     1975
## 132       FLASH FLOOD     1792
## 858          WILDFIRE     1456
## 204              HAIL     1361
```

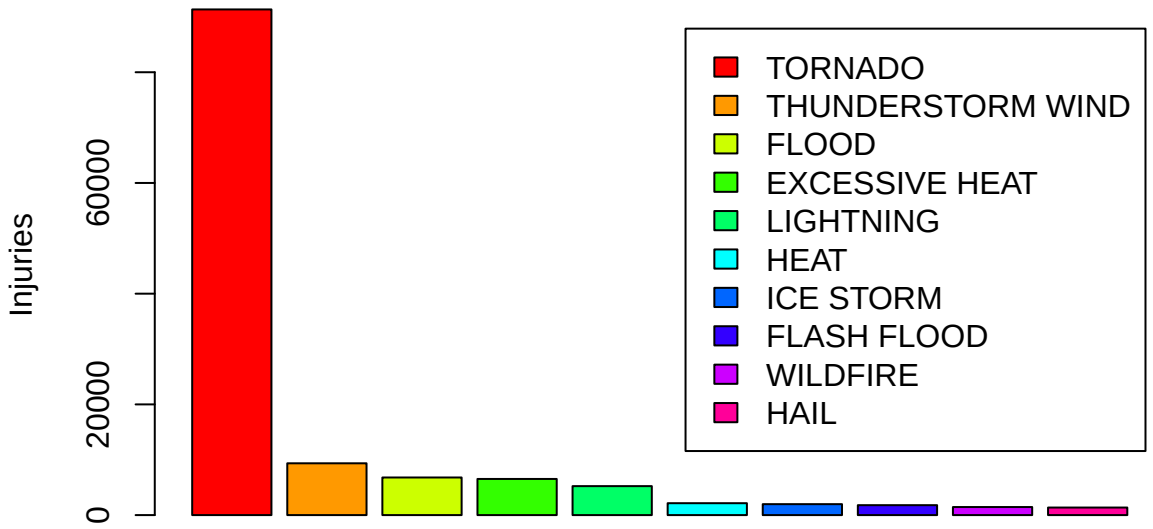
```r
# 6.1.2. finding the worst fatalities
fatStats = aggregate(FATALITIES~EVTYPE, data = modStormData, sum)
# exclude the results with zero fatalities
fatStats = fatStats [fatStats$FATALITIES>0,]
# arrange the results in descending order to find worst 10 results
fatDescr = fatStats [order(fatStats$FATALITIES, decreasing = TRUE),]
head (fatDescr, n=10)
```

```
##                EVTYPE FATALITIES
## 744           TORNADO       5633
## 109    EXCESSIVE HEAT       1903
## 132       FLASH FLOOD        995
## 235              HEAT        939
## 405         LIGHTNING        816
## 675 THUNDERSTORM WIND        701
## 514       RIP CURRENT        572
## 148             FLOOD        472
## 313         HIGH WIND        248
## 12          AVALANCHE        224
```

```r
# 6.1.3. plot the graphs for 10 worst injuries and fatalities
barplot(fatDescr[1:10, 2], col = rainbow(10), legend.text = fatDescr[1:10, 1], ylab = "Fatality", main =
```




```r
barplot(injDescr[1:10, 2], col = rainbow(10), legend.text = injDescr[1:10, 1], ylab = "Injuries", main =
```




```
# 6.2. Across the United States, which types of events have the greatest economic consequences?
```

```
# 6.2.1. Converting the Damages to absolute amounts
# find out the Exposure Symbols used for Property and Crop Damage Exponential
unique(c(unique(modStormData$PROPDMGEXP),unique(modStormData$CROPDMGEXP)))
```

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

```
# create the vector of the Exposure Symbols used for Property and Crop Damage Exponential
ExpSymb = c("", "+", "-", "?", 0:9, "H", "K", "M", "B")
# create the vector of the Exponential Power corresponding to each Exponential Symbols
ExpMultiplier = c(0, 0, 0, 0, 0:9, 2, 3, 6, 9)
# create a data frame combining Exponential Symbols and their multiplying powers
ExpPower = data.frame (ExpSymb, ExpMultiplier)
# calculate the actual Property Damage Amount using the Damage Amount and Exponential Symbol's Power
modStormData$PROPDMG = modStormData$PROPDMG * 10 ^ ExpPower[match(modStormData$PROPDMGEXP, ExpPower$Exp
# calculate the actual Crop Damage Amount using the Damage Amount and Exponential Symbol's Power
modStormData$CROPDMG = modStormData$CROPDMG * 10 ^ ExpPower[match(modStormData$CROPDMGEXP, ExpPower$Exp
# calculate the actual Total Damage Amount as sum of Property Damage and Crop Damage Amounts
modStormData$TotalDamage = modStormData$PROPDMG + modStormData$CROPDMG

# 6.2.2. finding the worst Property Damages
PropDamage = aggregate(PROPDMG~EVTYPE, data = modStormData, sum)
# exclude the results with zero fatalities
PropDamage = PropDamage [PropDamage$PROPDMG>0,]
# quantify the results in billions
PropDamage$PROPDMG = PropDamage$PROPDMG / 1000000000
# arrange the results in descending order to find worst 10 results
PropDamage = PropDamage[order(PropDamage$PROPDMG, decreasing = TRUE),]
head (PropDamage, n=10)
```

```
##                EVTYPE    PROPDMG
```

```
## 148             FLOOD 149.776655
## 364 HURRICANE-TYPHOON  81.174159
## 744           TORNADO  56.947381
## 587       STORM SURGE  43.323536
## 132       FLASH FLOOD  16.996735
## 204              HAIL  15.735268
## 675 THUNDERSTORM WIND   9.912672
## 858          WILDFIRE   7.766944
## 758    TROPICAL STORM   7.703891
## 874      WINTER STORM   6.688497
```

```
# 6.2.3. finding the worst Crop Damages
CropDamage = aggregate(CROPDMG~EVTYPE, data = modStormData, sum)
# exclude the results with zero fatalities
CropDamage = CropDamage [CropDamage$CROPDMG>0,]
# quantify the results in billions
CropDamage$CROPDMG = CropDamage$CROPDMG / 1000000000
# arrange the results in descending order to find worst 10 results
CropDamage = CropDamage[order(CropDamage$CROPDMG, decreasing = TRUE),]
head (CropDamage, n=10)
```

```
##                EVTYPE    CROPDMG
## 77            DROUGHT 13.9725660
## 148             FLOOD 10.6914275
## 364 HURRICANE-TYPHOON  5.3497828
## 380         ICE STORM  5.0221135
## 204              HAIL  3.0259545
## 179      FROST/FREEZE  1.5509110
## 132       FLASH FLOOD  1.5163511
## 118      EXTREME COLD  1.3129730
## 675 THUNDERSTORM WIND  1.1595052
## 246        HEAVY RAIN  0.7418879
```

```
# 6.2.4. finding the worst overall Damages
EconomicDamage = aggregate(TotalDamage~EVTYPE, data = modStormData, sum)
# exclude the results with zero fatalities
EconomicDamage = EconomicDamage [EconomicDamage$TotalDamage>0,]
# quantify the results in billions
EconomicDamage$TotalDamage = EconomicDamage$TotalDamage / 1000000000
# arrange the results in descending order to find worst 10 results
EconomicDamage = EconomicDamage[order(EconomicDamage$TotalDamage, decreasing = TRUE),]
head (EconomicDamage, n=10)
```

```
##                EVTYPE TotalDamage
## 148             FLOOD  160.468083
## 364 HURRICANE-TYPHOON   86.523942
## 744           TORNADO   57.362334
## 587       STORM SURGE   43.323541
## 204              HAIL   18.761222
## 132       FLASH FLOOD   18.513086
## 77            DROUGHT   15.018672
## 675 THUNDERSTORM WIND   11.072177
## 380         ICE STORM    8.967041
## 758    TROPICAL STORM    8.382237
```

```r
# 6.2.4. plotting the graphs for the results above
#barplot(PropDamage[1:10, 2], col = rainbow(10), legend.text = PropDamage[1:10, 1], ylab = "Property Da

#barplot(CropDamage[1:10, 2], col = rainbow(10), legend.text = CropDamage[1:10, 1], ylab = "Crop Damage

barplot(EconomicDamage[1:10, 2], col = rainbow(10), legend.text = EconomicDamage[1:10, 1], ylab = "Total
```