Summary of R Code for the NTSB Aviation Accidents Analysis

Jay Hill

February 3, 2016

This file is a streamlined complitation of the NTSB Aviation Accidents analysis. The full, rough code generated during exploratory analysis can be found in the accompanying files:

- `1 - load and clean.R`
- `2 - explore and transform.R`
- `3 - text analysis.R`
- `4 - full analysis.R`

Import the Raw Data Files

Start by loading all the packages needed for the analysis.

```
packs <- c("xml2", "readr", "jsonlite", "dplyr", "magrittr", "tm", "stringi", "svdvis",
"cluster", "ggplot2", "scales", "ggthemes", "lubridate", "reshape2")
sapply(packs, require, character.only = T)
```

```
##      xml2     readr  jsonlite     dplyr  magrittr        tm   stringi
##      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE
##    svdvis   cluster   ggplot2    scales  ggthemes lubridate  reshape2
##      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE      TRUE
```

Load the Aviation Data XML file, explore its structure, and extract the data into a data frame.

```
doc <- read_xml("data\\AviationData.xml")
rows <- xml_children(xml_children(doc))
numrows <- length(rows)
vars <- names(xml_attrs(rows[1])[[1]])
numvars <- length(vars)
events <- unlist(xml_attrs(rows)) %>%
    matrix(nrow=numrows, byrow=T) %>%
    data.frame(stringsAsFactors = F)
names(events) <- vars
```

As needed, convert the variables to more relevant types.

```
events$InvestigationType <- as.factor(events$InvestigationType)
events$EventDate <- as.Date(events$EventDate, "%m/%d/%Y")
events$Latitude <- as.numeric(events$Latitude)
events$Longitude <- as.numeric(events$Longitude)
events$AircraftDamage <- as.factor(events$AircraftDamage)
events$AircraftCategory <- as.factor(events$AircraftCategory)
events$AmateurBuilt <- as.factor(events$AmateurBuilt)
events$NumberOfEngines <- as.numeric(events$NumberOfEngines)
events$EngineType <- as.factor(events$EngineType)
events$FARDescription <- as.factor(events$FARDescription)
events$Schedule <- as.factor(events$Schedule)
events$PurposeOfFlight <- as.factor(events$PurposeOfFlight)
events$TotalFatalInjuries <- as.numeric(events$TotalFatalInjuries)
events$TotalSeriousInjuries <- as.numeric(events$TotalSeriousInjuries)
events$TotalMinorInjuries <- as.numeric(events$TotalMinorInjuries)
events$TotalUninjured <- as.numeric(events$TotalUninjured)
events$WeatherCondition <- as.factor(events$WeatherCondition)
events$BroadPhaseOfFlight <- as.factor(events$BroadPhaseOfFlight)
events$ReportStatus <- as.factor(events$ReportStatus)
events$PublicationDate <- as.Date(events$PublicationDate, "%m/%d/%Y")
```

Load the JSON files containing narrative and probable cause descriptions. There are 144 of these files, so cycle through them and build up the data frame by appending the new data at each step.

```
jsonFile <- "data\\NarrativeData_000.json"
jsonData <- fromJSON(jsonFile)
names(jsonData)
```

```
## [1] "data"
```

```
names(jsonData[[1]])
```

```
## [1] "EventId"        "narrative"      "probable_cause"
```

```
narratives <- jsonData[[1]]
for (i in seq(from = 499, to = 70999, by = 500)) {
    jsonFile <- paste0("data\\NarrativeData_", i, ".json")
    jsonData <- fromJSON(jsonFile)
    narratives <- rbind(narratives, jsonData[[1]])
}
jsonFile <- "data\\NarrativeData_999999.json"
jsonData <- fromJSON(jsonFile)
narratives <- rbind(narratives, jsonData[[1]])
```

Clean the Data

Check the date variables for inconsistencies. Then do the same for the latitude and longitude variables.

```
tail(events[c(3, 4, 31)], 20)
```

```
##       AccidentNumber  EventDate PublicationDate
## 77238     FTW82FPG08 1982-01-02      1983-01-02
## 77239     DEN82DTM08 1982-01-02      1983-01-02
## 77240     CHI82FEC08 1982-01-02      1983-01-02
## 77241     CHI82DA020 1982-01-02      1983-01-02
## 77242     ATL82DA027 1982-01-02      1983-01-02
## 77243     ANC82FAG14 1982-01-02      1983-01-02
## 77244     SEA82DA022 1982-01-01      1982-01-01
## 77245     NYC82DA015 1982-01-01      1982-01-01
## 77246     MIA82DA029 1982-01-01      1982-01-01
## 77247     FTW82DA034 1982-01-01      1982-01-01
## 77248     ATL82DKJ10 1982-01-01      1982-01-01
## 77249     CHI81LA106 1981-08-01      2001-11-06
## 77250     CHI79FA064 1979-08-02      1980-04-16
## 77251     LAX96LA321 1977-06-19      2000-09-12
## 77252     NYC07LA005 1974-08-30      2007-02-26
## 77253     LAX94LA336 1962-07-19      1996-09-19
## 77254     SEA87LA080 1948-10-24            <NA>
## 77255     WPR12TA445       <NA>      2013-02-08
## 77256     DCA00WA063       <NA>      2001-07-12
## 77257     CEN15FA325       <NA>      2015-08-10
```

```
    # observations 77251:77254 have EventDate inconsistent with AccidentNumber and
    # PublicationDate -- change these EventDate values to missing.
events$EventDate[77251:77254] <- NA
which(events$EventDate > events$PublicationDate)
```

```
## [1] 38376 40126
```

```
    # There are two cases where PublicationDate is before EventDate.
    # Mark these PublicationDate values to missing.
events$PublicationDate[c(38376, 40126)] <- NA

summary(events[7:8])
```

```
##     Latitude        Longitude
##  Min.   :-78.02   Min.   :-193.22
##  1st Qu.: 33.42   1st Qu.:-115.13
##  Median : 38.19   Median : -94.66
##  Mean   : 37.74   Mean   : -93.81
##  3rd Qu.: 42.57   3rd Qu.: -81.76
##  Max.   : 89.22   Max.   : 177.56
##  NA's   :53496    NA's   :53505
```

```
which(events$Longitude < -180)
```

```
## [1] 2803
```

```
    # one event with out of bounds longitude. Mark it missing.
events$Longitude[2803] <- NA
```

Impute missing values for the injuries variables where it makes sense to do so. For non-fatal crashes, set missing Total and Fraction fatal to 0. For serious and minor injuries, assume NA = 0. For total uninjured NA's, we don't have enough information to logically impute anything.

```
select(events, InjurySeverity, TotalFatalInjuries:TotalUninjured) %>%
    filter(InjurySeverity == "Non-Fatal") %>% summary
```

```
##  InjurySeverity     TotalFatalInjuries TotalSeriousInjuries
##  Length:58499       Min.   :0          Min.   : 0.000
##  Class :character   1st Qu.:0          1st Qu.: 0.000
##  Mode  :character   Median :0          Median : 0.000
##                     Mean   :0          Mean   : 0.285
##                     3rd Qu.:0          3rd Qu.: 0.000
##                     Max.   :0          Max.   :22.000
##                     NA's   :19730      NA's   :16789
##  TotalMinorInjuries TotalUninjured
##  Min.   :  0.000    Min.   :  0.000
##  1st Qu.:  0.000    1st Qu.:  1.000
##  Median :  0.000    Median :  1.000
##  Mean   :  0.525    Mean   :  3.878
##  3rd Qu.:  1.000    3rd Qu.:  2.000
##  Max.   :200.000    Max.   :699.000
##  NA's   :15290      NA's   :5103
```

```
events[events$InjurySeverity == "Non-Fatal", "TotalFatalInjuries"] <- 0
events[is.na(events$TotalSeriousInjuries), "TotalSeriousInjuries"] <- 0
events[is.na(events$TotalMinorInjuries), "TotalMinorInjuries"] <- 0
```

Fill in missing NumberOfEngines values by using values from the same model of aircraft. First, create a list of unique models and their number of engines to use as a look-up table. Next, find the observations with missing number of engine values. Finally, fill in the missing values if a suitable value is available in the look-up table. For any remaing missing values, set the Number of Engines eqaul to 1 as that is by far the most likely value in the data set.

```
events$Model <- toupper(events$Model)

ModelAndEngines <-select(events, Model, NumberOfEngines) %>%
    filter(!is.na(Model)) %>% filter(!is.na(NumberOfEngines))
ModelAndEngines$Model <- removePunctuation(ModelAndEngines$Model) %>%
    stripWhitespace %>%
    stri_trim_both
ModelAndEngines <- unique(ModelAndEngines) %>%
    arrange(Model, desc(NumberOfEngines))
ModelAndEngines <- ModelAndEngines[-which(duplicated(ModelAndEngines$Model)), ]
ModelAndEngines <- ModelAndEngines[-1, ]  # first row has " " model
missingEngines <- which(is.na(events$NumberOfEngines))
for (i in missingEngines) {
    tempModel <- events[i, "Model"]
    tempModel <- removePunctuation(tempModel) %>% stripWhitespace %>%
        stri_trim_both
    tempNumEng <- ModelAndEngines[ModelAndEngines$Model == tempModel, 2]
    if (tempModel %in% ModelAndEngines$Model) {
        events$NumberOfEngines[i] <- tempNumEng
    }
}
summary(events$NumberOfEngines)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   1.000   1.000   1.157   1.000  24.000     806
```

```
events[is.na(events$NumberOfEngines), "NumberOfEngines"] <- 1
```

Perform some more housekeeping on the variables. Some factor variables have empty string for a label; replace as “(missing)”. Other character variables have “” values; change to “(missing)”.

```
levels(events$AircraftDamage)[1] <- "(missing)"
levels(events$AircraftCategory)[1] <- "(missing)"
levels(events$AmateurBuilt)[1] <- "(missing)"
levels(events$EngineType)[1] <- "(missing)"
levels(events$FARDescription)[1] <- "(missing)"
levels(events$Schedule)[1] <- "(missing)"
levels(events$PurposeOfFlight)[1] <- "(missing)"
levels(events$WeatherCondition)[1] <- "(missing)"
levels(events$BroadPhaseOfFlight)[1] <- "(missing)"

events[events$Location == "", "Location"] <- "(missing)"
events[events$Country == "", "Country"] <- "(missing)"
events[events$AirportCode == "", "AirportCode"] <- "(missing)"
events[events$AirportName == "", "AirportName"] <- "(missing)"
events[events$InjurySeverity == "", "InjurySeverity"] <- "(missing)"
events[events$RegistrationNumber == "", "RegistrationNumber"] <- "(missing)"
events[events$Make == "", "Make"] <- "(missing)"
events[events$Model == "", "Model"] <- "(missing)"
events[events$AirCarrier == "", "AirCarrier"] <- "(missing)"
```

Create New Variables

In addition to the totals provided in the data set, it will be useful to consider fractions of passengers receiving injuries. Create a new variable for total passengers, then use that value to create variables for fraction of passengers killed, injured, or uninjured.

```
passengers <- events[, c("TotalFatalInjuries", "TotalSeriousInjuries",
                         "TotalMinorInjuries", "TotalUninjured")]
events$TotalPassengers <- rowSums(passengers, na.rm = T)
# If all four categories are missing, set TotalPassengers as missing.
# Otherwise assume TotalPassengers is the sum of the non-missing categories.
allmissing <- rowSums(is.na(passengers)) == 4
events$TotalPassengers <- events$TotalPassengers * ifelse(allmissing, NA, 1)
summary(events$TotalPassengers)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   6.008   2.000 699.000
```

```
events$FractionFatalInjuries <- events$TotalFatalInjuries /
    events$TotalPassengers
events$FractionSeriousInjuries <- events$TotalSeriousInjuries /
    events$TotalPassengers
events$FractionMinorInjuries <- events$TotalMinorInjuries /
    events$TotalPassengers
events$FractionUninjured <- events$TotalUninjured / events$TotalPassengers
```

Text Analysis and Clustering

Clean the narrative and probable cause description text and prepare it for analysis. This function creates clean, single-word tokens. Punctuation is removed, changed to all lower-case, extra white space is removed, and common stop words are removed.

Since the report does not discuss the probable cause text, this document will omit it as well. Refer to `3 - text analysis.R` for the full analysis.

```
cleaner <- function(txt) {
    output <- txt %>%
        stri_trans_tolower %>%
        removeWords(stopwords(kind = "en")) %>%
        stri_replace_all_fixed("--", " ") %>%
        removePunctuation %>%
        stripWhitespace %>%
        stri_trim_both
    return(output)
}

narratives$narrative <- cleaner(narratives$narrative)
```

Separate the narratives and probable causes into 2 data frames, then remove rows with no narrative.

```
probable_causes <- narratives
narratives$probable_cause <- NULL
narratives <- narratives[stri_length(narratives$narrative) > 0, ]
```

Create a term-document matrix for the narratives. Weight the TDM using term frequency - inverse document frequency. Remove sparse terms to focus on only the most frequently appearing terms.

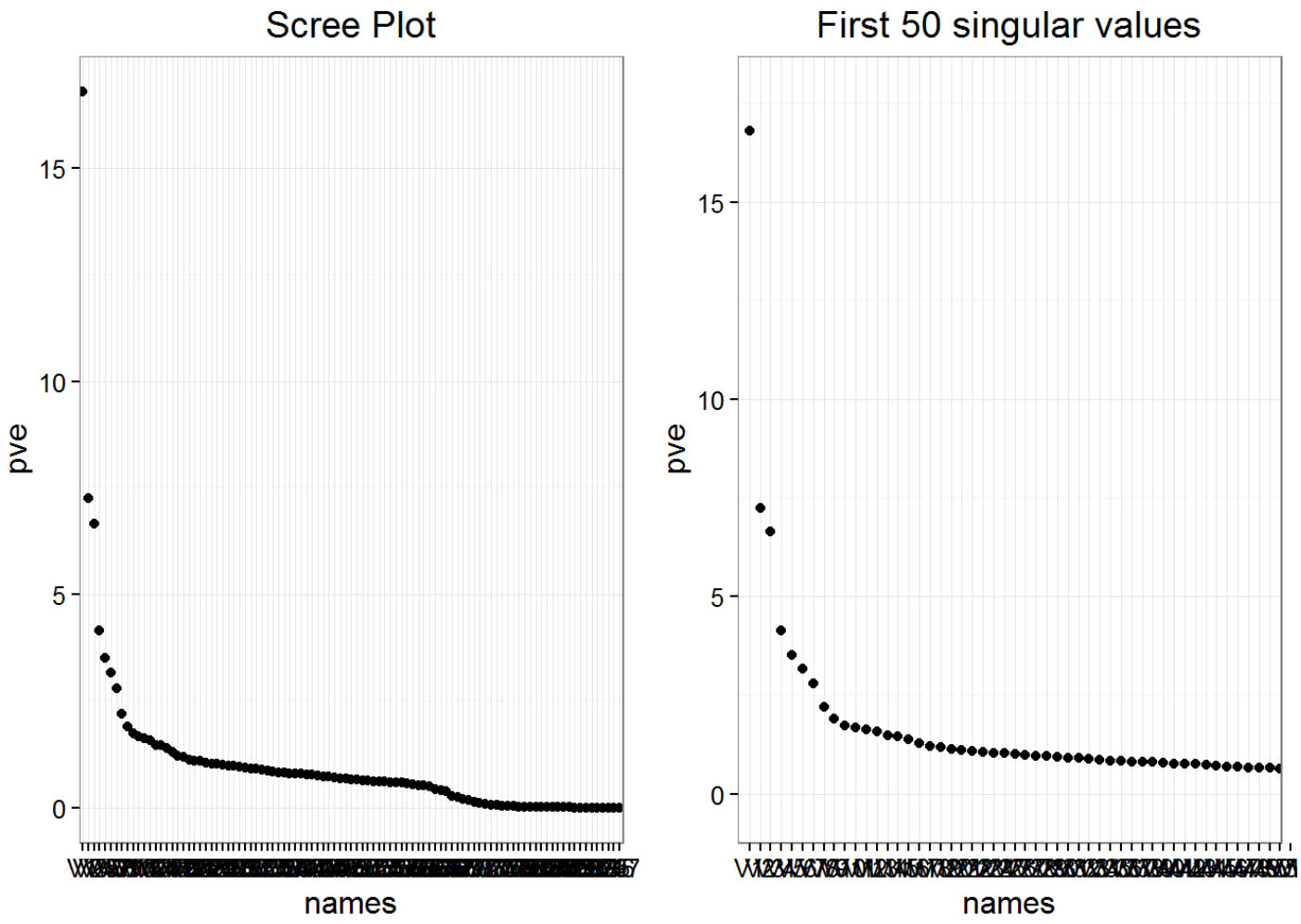
```
TDMer <- function(txt, sparsity) {
    output <- VectorSource(txt) %>%
        VCorpus %>%
        TermDocumentMatrix %>%
        weightTfIdf %>%
        removeSparseTerms(sparse = sparsity)
    return(output)
}

narrativeTDM <- TDMer(narratives$narrative, 0.90)  # 97 terms remain
```

Weight the term-document matrix by using singular value decomposition to perform latent semantic analysis.

```
narrativeMatrix <- as.matrix(narrativeTDM)
dimnames(narrativeMatrix)[[2]] <- narratives$EventId
narrativeSVD <- svd(narrativeMatrix)
svd.scree(narrativeSVD, subr = 50)  # elbow is around 9 singular values
```

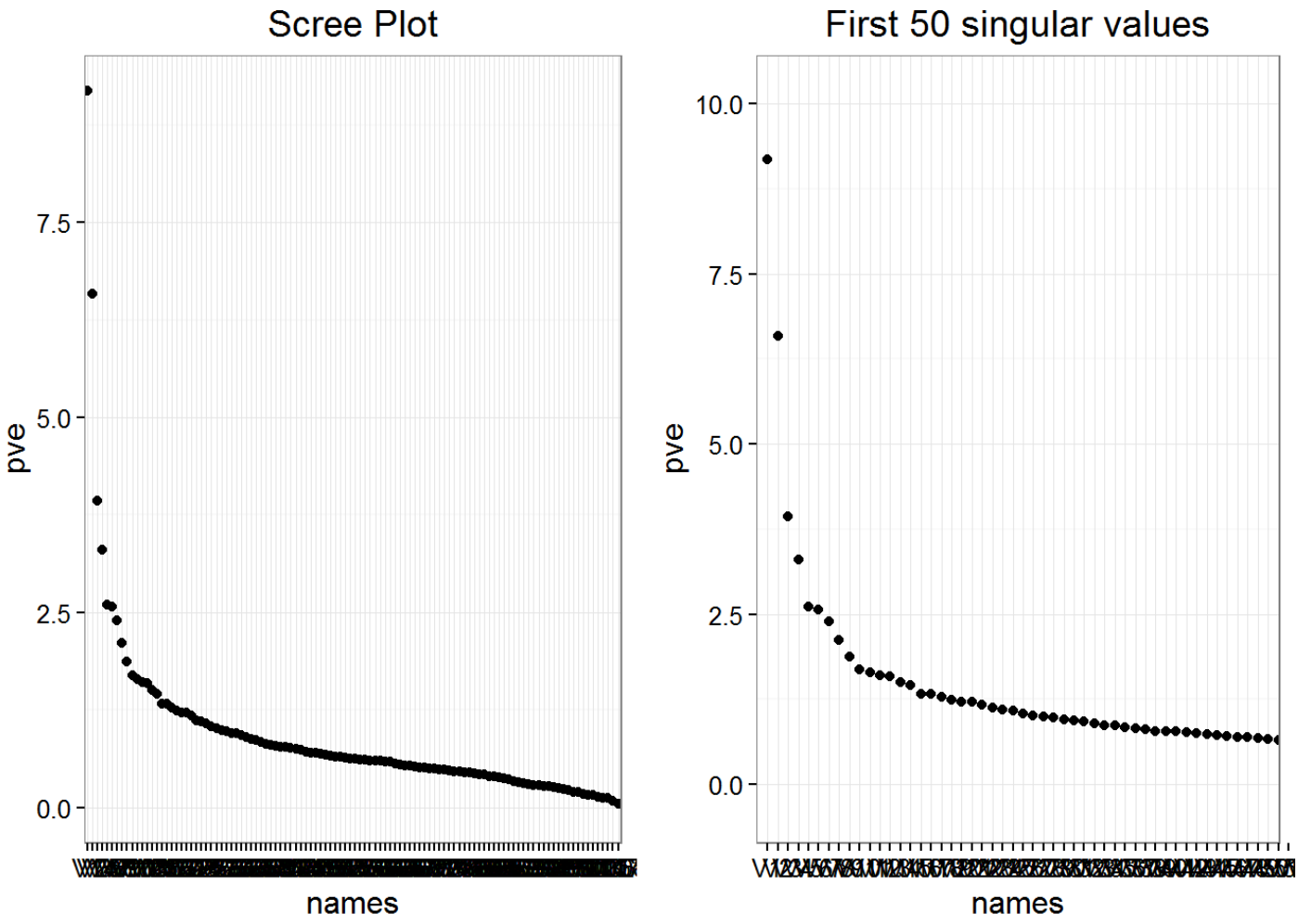
```
## [1] "Your input data is treated as a SVD output, with u, d, v corresponding to left si
ngular vector, singular values, and right singular vectors, respectively."
## [1] "Scree Plot"
```




```
## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

```
singvals <- 9
U <- narrativeSVD$u[, 1:singvals]
D <- diag(narrativeSVD$d[1:singvals])
Vt <- t(narrativeSVD$v[, 1:singvals])
prin1nar <- Vt[1, ]
prin2nar <- Vt[2, ]
narrativeLSA <- U %*% D %*% Vt
dimnames(narrativeLSA) <- dimnames(narrativeMatrix)
```

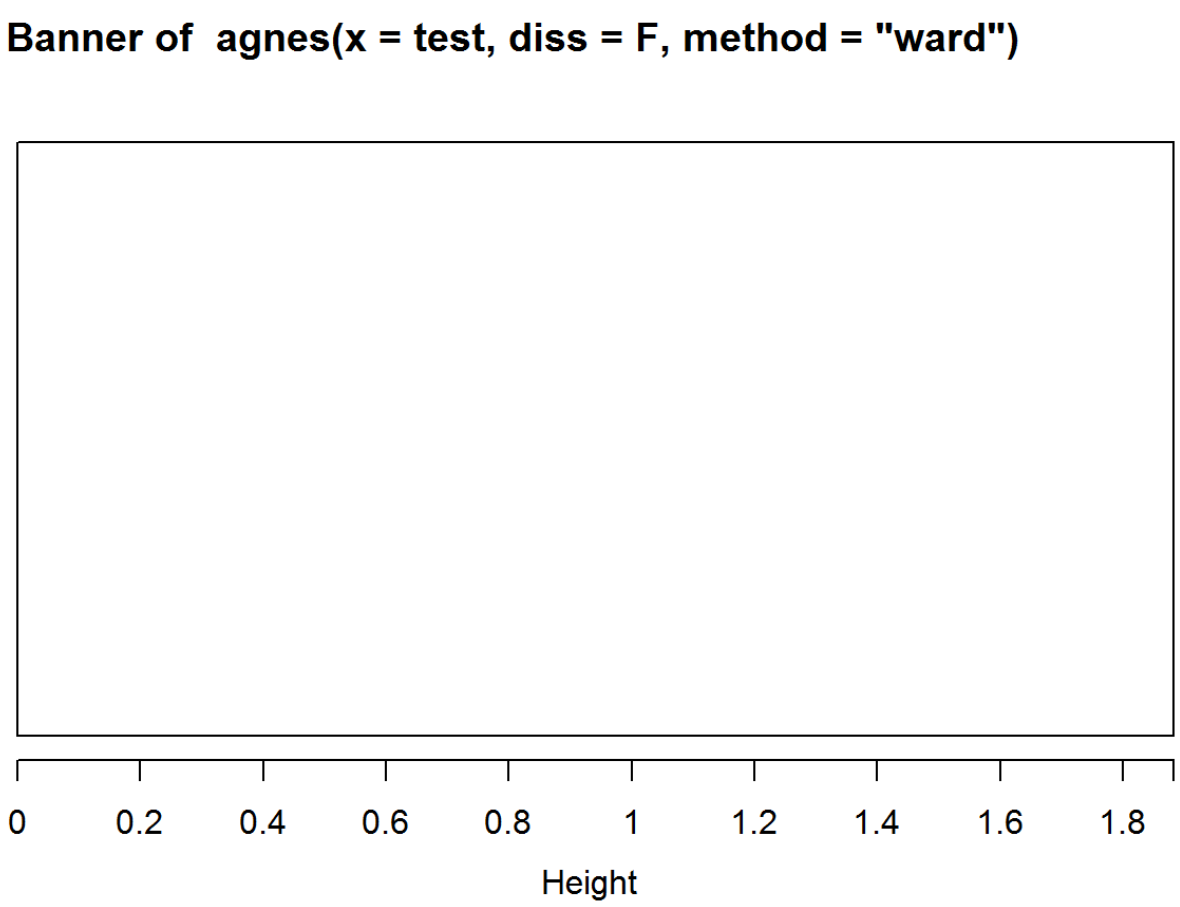
```
## [1] "Your input data is treated as a SVD output, with u, d, v corresponding to left si
ngular vector, singular values, and right singular vectors, respectively."
## [1] "Scree Plot"
```

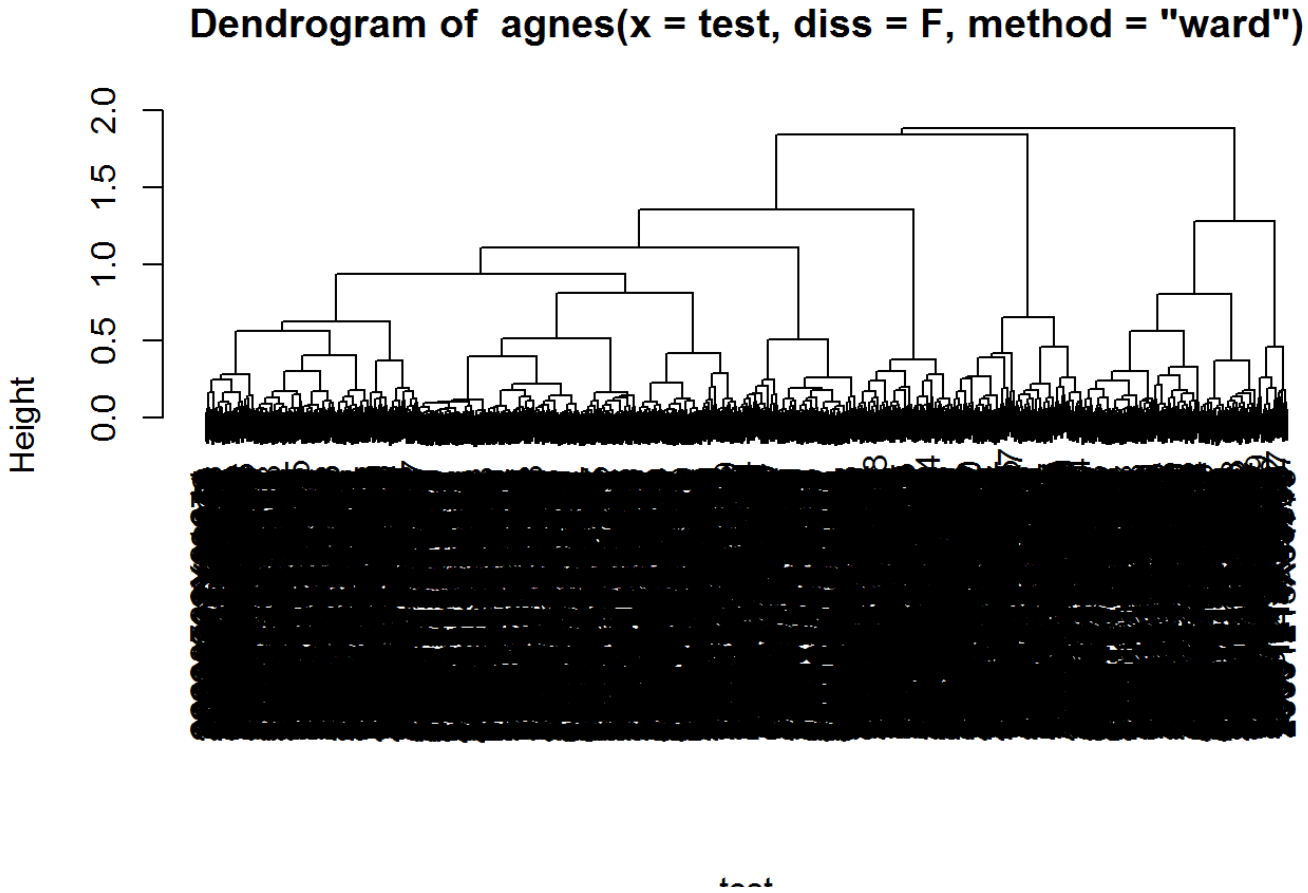



```
## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

Perform hierarchical clustering to gain an estimate for the number of clusters to use in the non-hierarchical clustering method. Due to memory constraints, do this analysis on a sample of the data.

```
hierclust <- function(LSA) {
    set.seed(5678)
    nobs <- ncol(LSA)
    testSample <- sample(x = 1:nobs, size = 1000)
    test <- t(LSA[, testSample])
    output <- agnes(test, diss = F, method = "ward")
    return(output)
}
narAGNES <- hierclust(narrativeLSA)
plot(narAGNES)
```


Banner of agnes(x = test, diss = F, method = "ward")
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
Height
Agglomerative Coefficient = 0.98

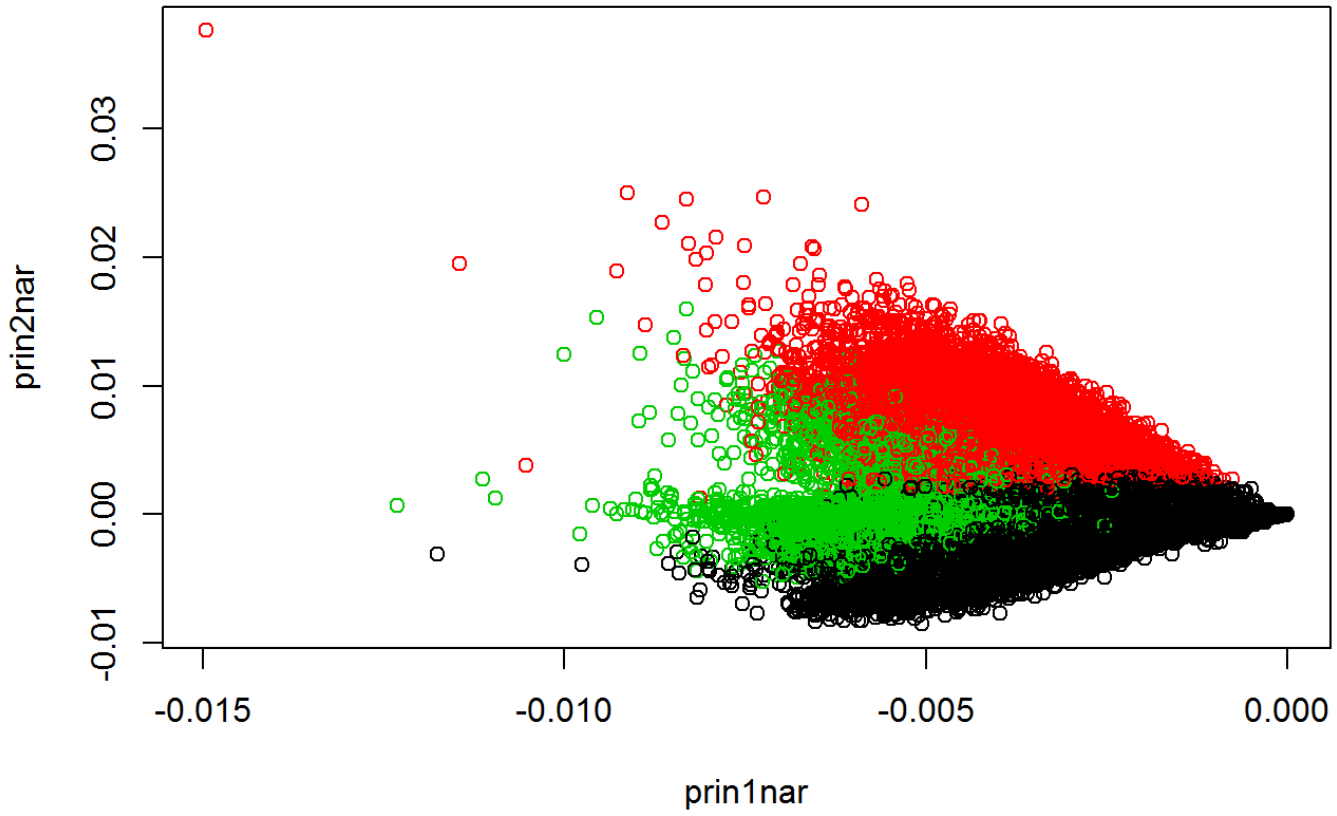

Dendrogram of agnes(x = test, diss = F, method = "ward")
Height
0.0
0.5
1.0
1.5
2.0
test

test
Agglomerative Coefficient = 0.98

From the tree, it looks like 3, 5, and 6 are good candidates for the number of clusters.

Perform non-hierarchical using the CLARA method, which uses repeated samples drawn from the full data set. Then plot the scores along the first two principal components.

```
set.seed(1234)
narrative3Clusters <- clara(t(narrativeLSA), k = 3, metric = "euclidean",
                            samples = 50, sampsize = 1000, rngR = T)
plot(prin1nar, prin2nar, col=narrative3Clusters$cluster)
```




```
narrative3Clusters$clusinfo
```

```
##       size  max_diss    av_diss isolation
## [1,] 55135 0.5207755 0.07980380  4.358317
## [2,] 12373 0.6250649 0.08536381  5.231104
## [3,]  8397 0.4188845 0.08446771  3.429065
```

Interpret the clusters by looking at the top twenty highest weighted terms in each cluster.

```
narClusLSA <- as.data.frame(cbind(t(narrativeLSA),
                                  cluster=narrative3Clusters$clustering))
narTermMeans <- t(aggregate(narClusLSA, by=list(narClusLSA$cluster), FUN=mean))
narClus1Terms <- sort(narTermMeans[, 1], decreasing = T)
narClus2Terms <- sort(narTermMeans[, 2], decreasing = T)
narClus3Terms <- sort(narTermMeans[, 3], decreasing = T)
names(narClus1Terms)[3:22]
```

```
##  [1] "airplane" "runway"   "engine"   "gear"     "flight"   "landing"
##  [7] "pilot"    "left"     "right"    "reported" "feet"     "airport"
## [13] "power"    "ground"   "approach" "wind"     "wing"     "stated"
## [19] "nose"     "takeoff"
```

```
names(narClus2Terms)[3:22]
```

```
##  [1] "acft"          "plt"           "gear"          "takeoff"
##  [5] "stated"        "landing"       "obtained"      "left"
##  [9] "work"          "without"       "investigative" "nose"
## [13] "said"          "either"        "collided"      "amount"
## [17] "significant"   "wind"          "conducted"     "may"
```

```
names(narClus3Terms)[3:22]
```

```
##  [1] "fuel"        "engine"      "power"       "forced"      "airplane"
##  [6] "found"       "lost"        "loss"        "flight"      "revealed"
## [11] "landing"     "field"       "right"       "examination" "left"
## [16] "pilot"       "airport"     "stated"      "made"        "acft"
```

Here are some possible interpretations of the clusters.

1. Mechanical issues at the airport or during takeoff/landing. (runway, airport; landing, takeoff, approach; engine, gear, wing, nose)
2. Wind or collision related during takeoff/landing (takeoff, landing; gear, nose; collided; wind)
3. Loss of fuel or power during flight (fuel, engine, power; loss, lost, found; flight)

Create a data frame of the Event IDs and the cluster assignments. Merge this data frame with the ‘events’ data frame.

```
narrativeFinal <- data.frame(EventId = colnames(narrativeLSA),
                             NarrativeCluster=narrative3Clusters$clustering)
rownames(narrativeFinal) <- NULL
probable_causeFinal <- data.frame(EventId = colnames(probcauseLSA),
                          ProbableCauseCluster=probcause4Clusters$clustering)
rownames(probable_causeFinal) <- NULL
textFinal <- merge(x = narrativeFinal, y = probable_causeFinal, all = T)
events <- merge(x = events, y = textFinal, all.x = T)
```

Final Analysis of Combined Events and Narratives Data

Create some new date variables that are easier to work with.

```
# add month and year variables; sort by date
events <- arrange(events, EventDate) %>%
    mutate(EventMonth = cut(EventDate, breaks = "months"),
           EventYear = cut(EventDate, breaks = "years"))
    # Only 1 observation each in 1979 and 1981, seven missing dates; remove these
events <- events[3:(nrow(events)-7), ]
# add a variable for the name of the month
monthlevels = c("January", "February", "March", "April", "May", "June", "July",
                "August", "September", "October", "November", "December")
events$EventMonthName <- months(events$EventDate) %>%
    factor(levels = monthlevels)
# add a variable for the year as a four digit number
events <- mutate(events, EventYearOnly = year(EventYear))
```

Calculate some basic summary statistics and related information that were referenced in the report.

```
summary(events$EventYearOnly)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1982    1987    1995    1996    2004    2015
```

```
table(events$NarrativeCluster)
```

```
##
##     1     2     3
## 56052 12566  8399
```

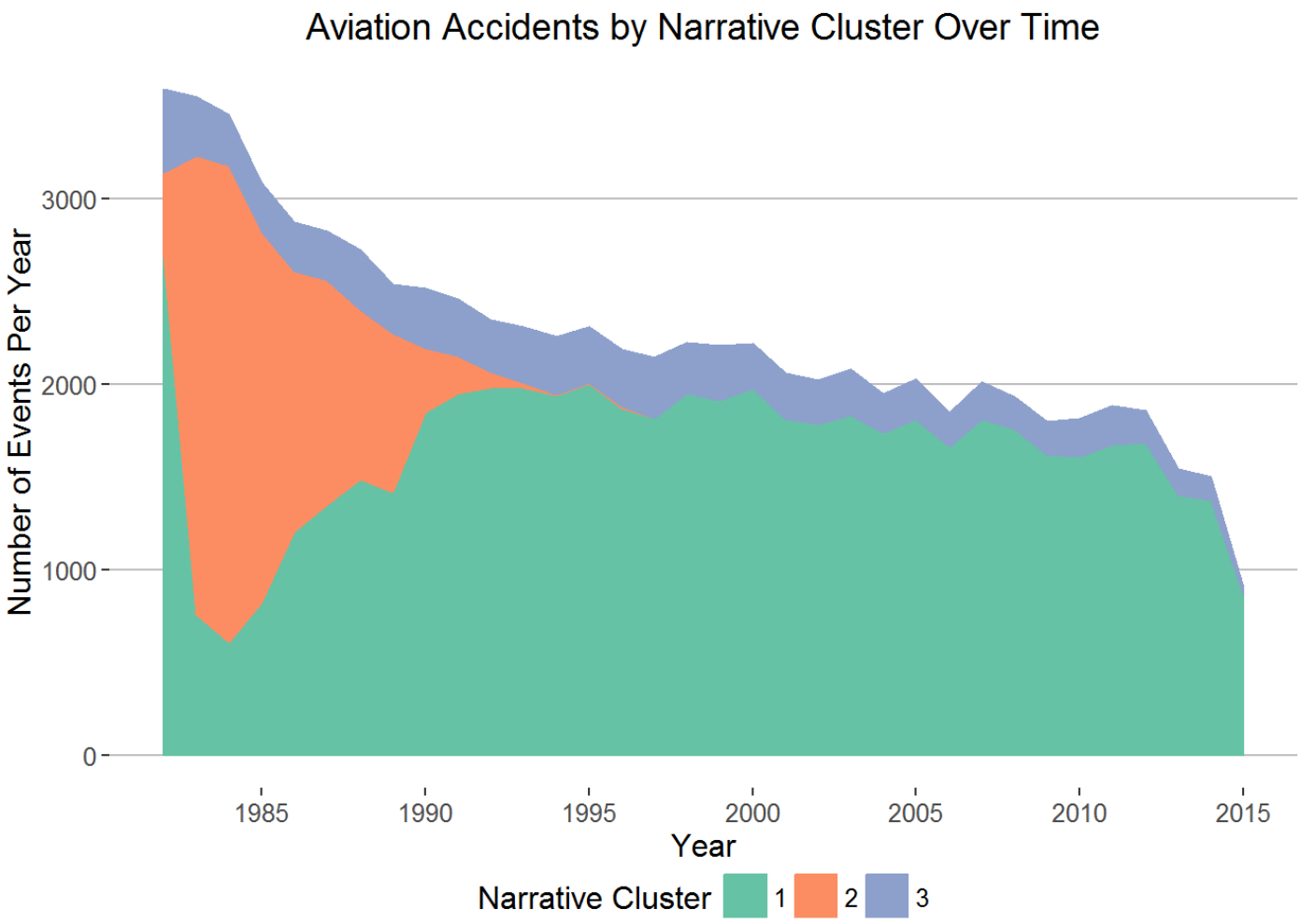
Figure A

Look at changes in narrative cluster prevalence over time.

```
eventsGrouped <- group_by(events, EventYear, NarrativeCluster)
# remove observations with missing narrative clusters (231 of 77,248)
eventsGrouped <- filter(eventsGrouped, !is.na(NarrativeCluster))
# summarize for each year and cluster type
(by_cluster <- summarise(eventsGrouped, numPerClusterPerYear = n()))
```

```
## Source: local data frame [84 x 3]
## Groups: EventYear [?]
##
##     EventYear NarrativeCluster numPerClusterPerYear
##        (fctr)            (int)                (int)
## 1  1982-01-01                1                 2688
## 2  1982-01-01                2                  455
## 3  1982-01-01                3                  446
## 4  1983-01-01                1                  760
## 5  1983-01-01                2                 2472
## 6  1983-01-01                3                  319
## 7  1984-01-01                1                  608
## 8  1984-01-01                2                 2575
## 9  1984-01-01                3                  268
## 10 1985-01-01                1                  824
## ..        ...              ...                  ...
```

```
ggAfinal <- ggplot(by_cluster, aes(x = as.Date(as.character(EventYear)),
                                   y = numPerClusterPerYear)) +
    geom_area(aes(colour = as.character(NarrativeCluster),
                  fill = as.character(NarrativeCluster))) +
    theme_hc() +
    theme(plot.title = element_text(size = 14),
          legend.margin = unit(0, "cm")) +
    scale_color_brewer(type = "qual", palette = 7, name = "Narrative Cluster") +
    scale_fill_brewer(type = "qual", palette = 7, name = "Narrative Cluster") +
    xlab("Year") +
    ylab("Number of Events Per Year") +
    ggtitle("Aviation Accidents by Narrative Cluster Over Time")
ggAfinal
```



The following tables describe the data presented in Figure A.

```
prop.table(table(events$NarrativeCluster, events$EventYearOnly), 2)
```

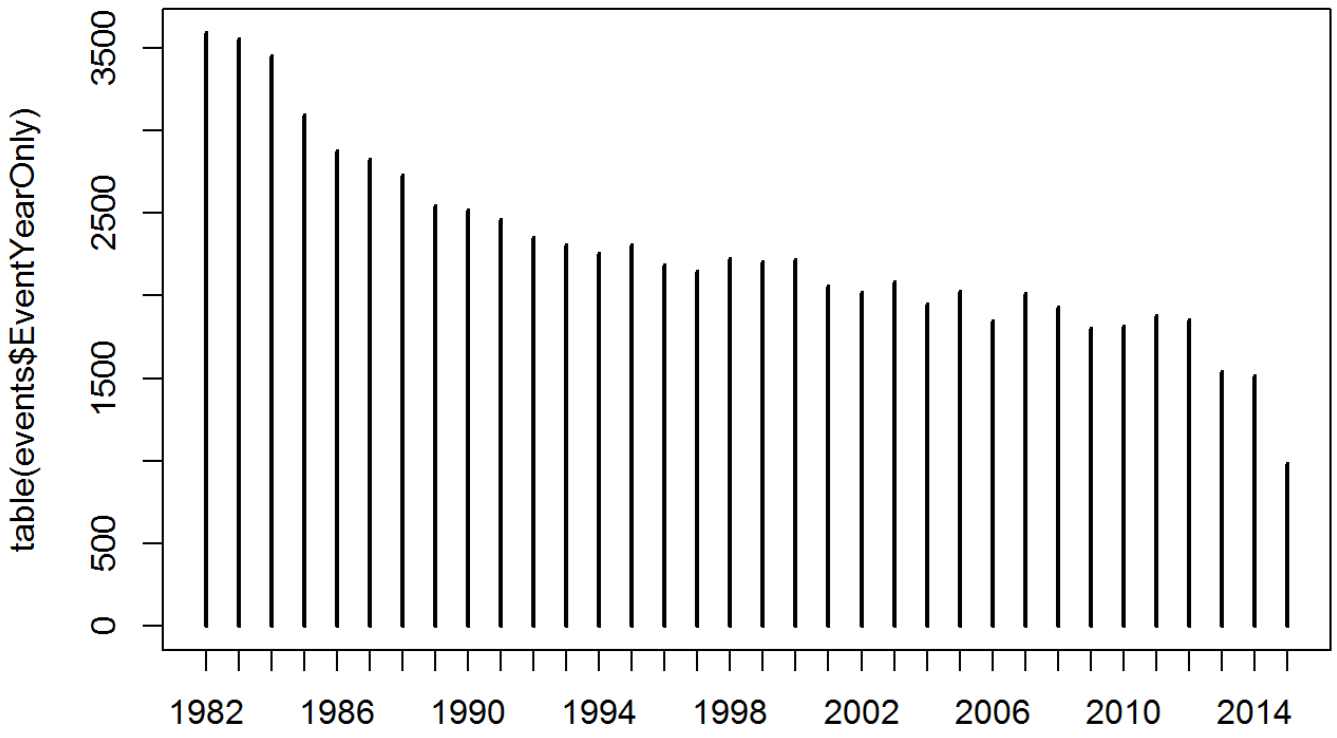
```
##
##            1982        1983        1984        1985        1986
##   1 0.748955141 0.214024219 0.176180817 0.267098865 0.420173913
##   2 0.126776261 0.696141932 0.746160533 0.647649919 0.487652174
##   3 0.124268598 0.089833850 0.077658650 0.085251216 0.092173913
##
##            1987        1988        1989        1990        1991
##   1 0.477876106 0.546221570 0.559101655 0.736381710 0.793411956
##   2 0.429380531 0.335656640 0.336879433 0.137972167 0.083367222
##   3 0.092743363 0.118121790 0.104018913 0.125646123 0.123220821
##
##            1992        1993        1994        1995        1996
##   1 0.845826235 0.860424794 0.860815603 0.869904597 0.856358646
##   2 0.035349233 0.011703511 0.001773050 0.001300954 0.004117109
##   3 0.118824532 0.127871695 0.137411348 0.128794449 0.139524245
##
##            1997        1998        1999        2000        2001
##   1 0.848272642 0.878487849 0.867180417 0.891696751 0.882410107
##   2 0.001400560 0.000000000 0.000000000 0.000000000 0.000000000
##   3 0.150326797 0.121512151 0.132819583 0.108303249 0.117589893
##
##            2002        2003        2004        2005        2006
##   1 0.883663366 0.881364073 0.892252437 0.893491124 0.901843818
##   2 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##   3 0.116336634 0.118635927 0.107747563 0.106508876 0.098156182
##
##            2007        2008        2009        2010        2011
##   1 0.900546992 0.911871436 0.900555556 0.887665198 0.890425532
##   2 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##   3 0.099453008 0.088128564 0.099444444 0.112334802 0.109574468
##
##            2012        2013        2014        2015
##   1 0.906788793 0.909208820 0.919893191 0.958934517
##   2 0.000000000 0.000000000 0.000000000 0.000000000
##   3 0.093211207 0.090791180 0.080106809 0.041065483
```

```
table(events$NarrativeCluster, events$EventYearOnly)
```

```
##
##     1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995
##   1 2688  760  608  824 1208 1350 1489 1419 1852 1951 1986 1985 1942 2006
##   2  455 2472 2575 1998 1402 1213  915  855  347  205   83   27    4    3
##   3  446  319  268  263  265  262  322  264  316  303  279  295  310  297
##
##     1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
##   1 1872 1817 1952 1913 1976 1816 1785 1835 1739 1812 1663 1811 1759 1621
##   2    9    3    0    0    0    0    0    0    0    0    0    0    0    0
##   3  305  322  270  293  240  242  235  247  210  216  181  200  170  179
##
##     2010 2011 2012 2013 2014 2015
##   1 1612 1674 1683 1402 1378  864
##   2    0    0    0    0    0    0
##   3  204  206  173  140  120   37
```

These are some additional supporting plots that show the mild exponential decrease in total events over the time period of interest.

```
plot(table(events$EventYearOnly))
```



```
plot(log(table(events$EventYearOnly)))
```

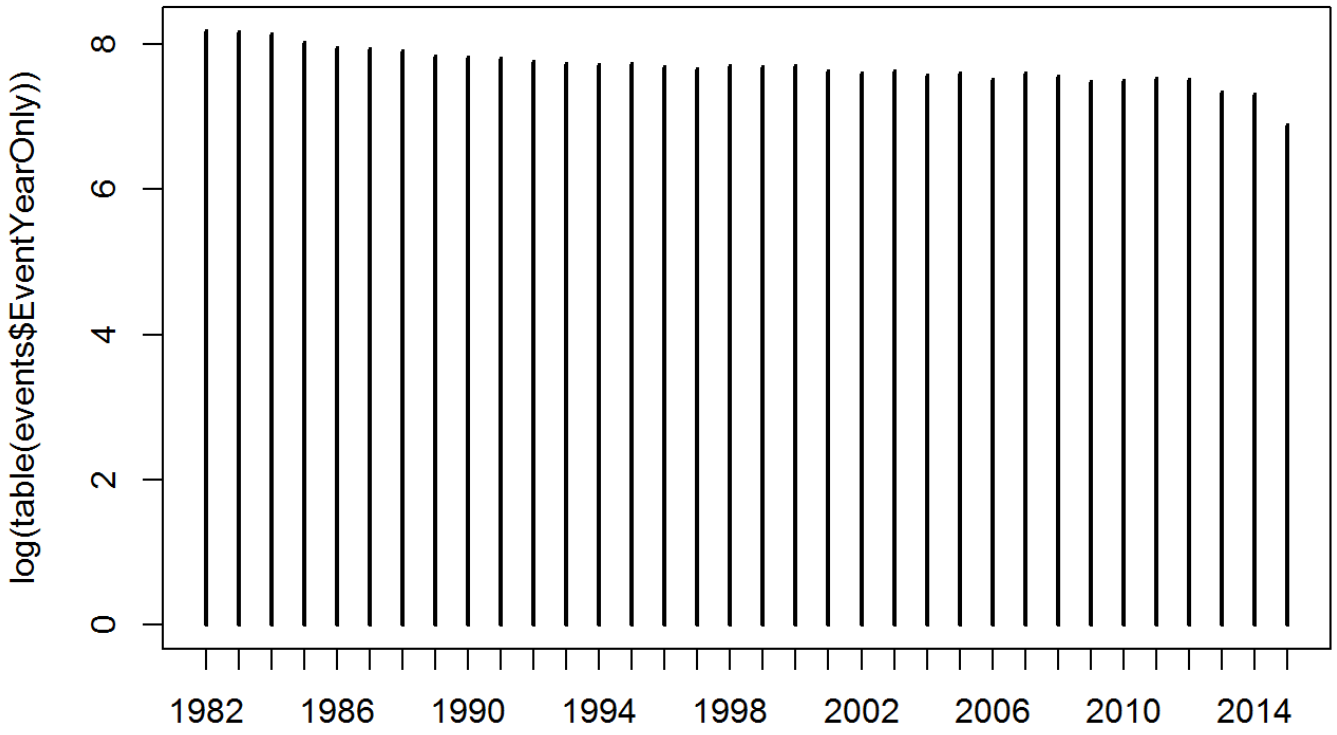
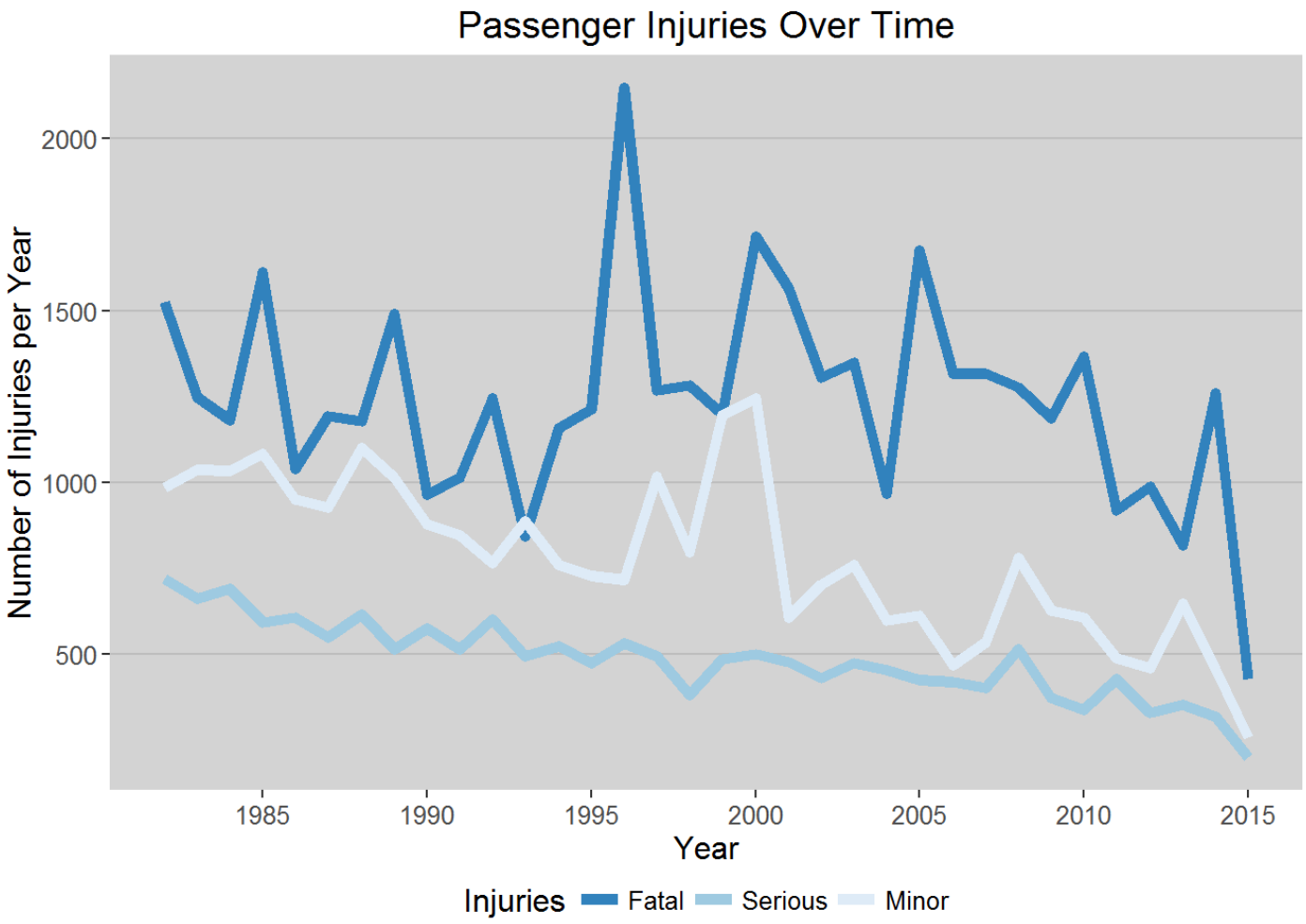


Figure B

Examine casualty counts over time.

```
eventsGrouped3 <- events[!duplicated(events$EventId), ] # to avoid double counts
eventsGrouped3 <- group_by(eventsGrouped3, EventYearOnly) %>%
    select(EventYearOnly, TotalFatalInjuries:TotalMinorInjuries) %>%
    summarise(FatalPerYear = sum(TotalFatalInjuries, na.rm=T),
              SeriousPerYear = sum(TotalSeriousInjuries, na.rm=T),
              MinorPerYear = sum(TotalMinorInjuries, na.rm=T)) %>%
    melt(id.vars = "EventYearOnly", variable.name = "Injuries",
         value.name = "NumberPerYear")
ggBfinal <- ggplot(eventsGrouped3, aes(x = EventYearOnly, y = NumberPerYear,
                                       group = Injuries, color = Injuries)) +
    geom_line(size = 2) +
    scale_x_continuous(breaks = seq(1985, 2015, 5)) +
    theme_hc() +
    theme(panel.background = element_rect(fill = 'lightgray'),
          legend.margin = unit(0, "cm")) +
    scale_color_brewer(type = "seq", palette = 1, direction = -1,
                       labels = c("Fatal", "Serious", "Minor")) +
    xlab("Year") +
    ylab("Number of Injuries per Year") +
    ggtitle("Passenger Injuries Over Time")
ggBfinal
```

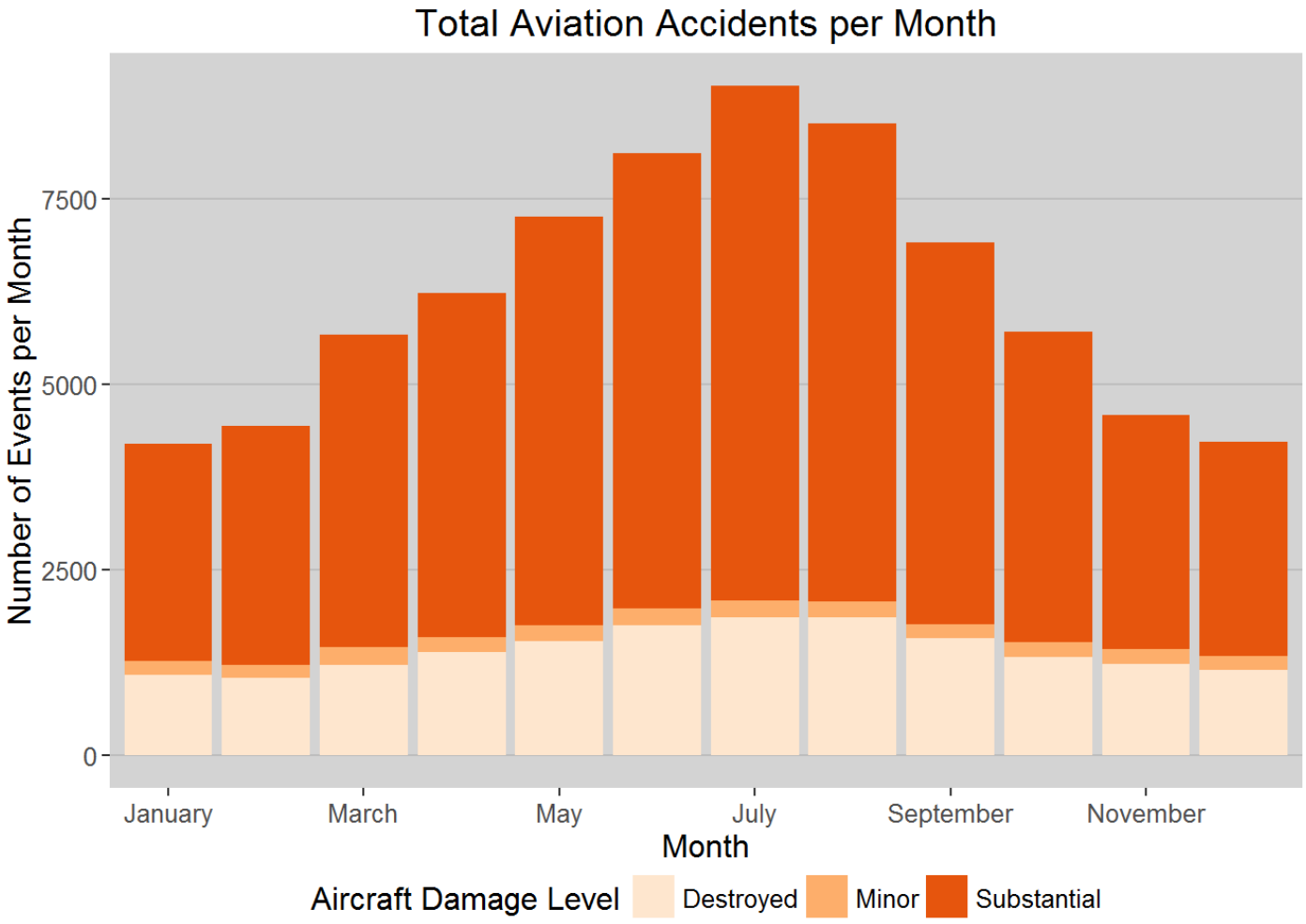


```
    # Fatal injuries are the most common, serious are least common
```

Figure C

Look at distributions of events within a year.

```
ggCfinal <- ggplot(filter(events, AircraftDamage != "(missing)"),
              aes(x = EventMonthName)) +
    geom_bar(aes(fill = AircraftDamage)) +
    scale_x_discrete(breaks = c("January", "March", "May", "July", "September",
                                "November")) +
    theme_hc() +
    theme(panel.background = element_rect(fill = 'lightgray'),
          legend.margin = unit(0, "cm")) +
    scale_fill_brewer(type = "seq", palette = 7,
                      name = "Aircraft Damage Level") +
    xlab("Month") +
    ylab("Number of Events per Month") +
    ggtitle("Total Aviation Accidents per Month")
ggCfinal
```



These are some supporting tables of the same data presented in the above figure.

```
eventsGrouped2 <- group_by(events, EventMonthName, AircraftDamage) %>%
    filter(AircraftDamage != "(missing)")
table(events$EventMonthName)
```

```
##
##   January  February     March     April       May      June      July
##      4345      4597      5883      6429      7479      8341      9306
##    August September   October  November  December
##      8747      7087      5917      4743      4374
```

```
100*prop.table(table(eventsGrouped2$AircraftDamage, eventsGrouped2$EventMonthName), 2)
```

```
##
##                 January  February     March     April       May      June
##   (missing)    0.000000  0.000000  0.000000  0.000000  0.000000  0.000000
##   Destroyed   25.871122 23.382187 21.369331 22.261427 21.130255 21.508690
##   Minor        4.439141  3.945885  4.199753  3.223737  3.059959  2.847282
##   Substantial 69.689737 72.671928 74.430916 74.514836 75.809786 75.644028
##
##                    July    August September   October  November  December
##   (missing)    0.000000  0.000000  0.000000  0.000000  0.000000  0.000000
##   Destroyed   20.643016 21.817755 22.802198 23.153658 26.834061 27.225379
##   Minor        2.461197  2.430719  2.747253  3.535177  4.279476  4.261364
##   Substantial 76.895787 75.751527 74.450549 73.311166 68.886463 68.513258
```

This stacked bar chart with proportional axis shows the fraction of aircraft suffering each level of damage in each month.

```
levels(events$AircraftDamage)
```

```
## [1] "(missing)"   "Destroyed"   "Minor"       "Substantial"
```

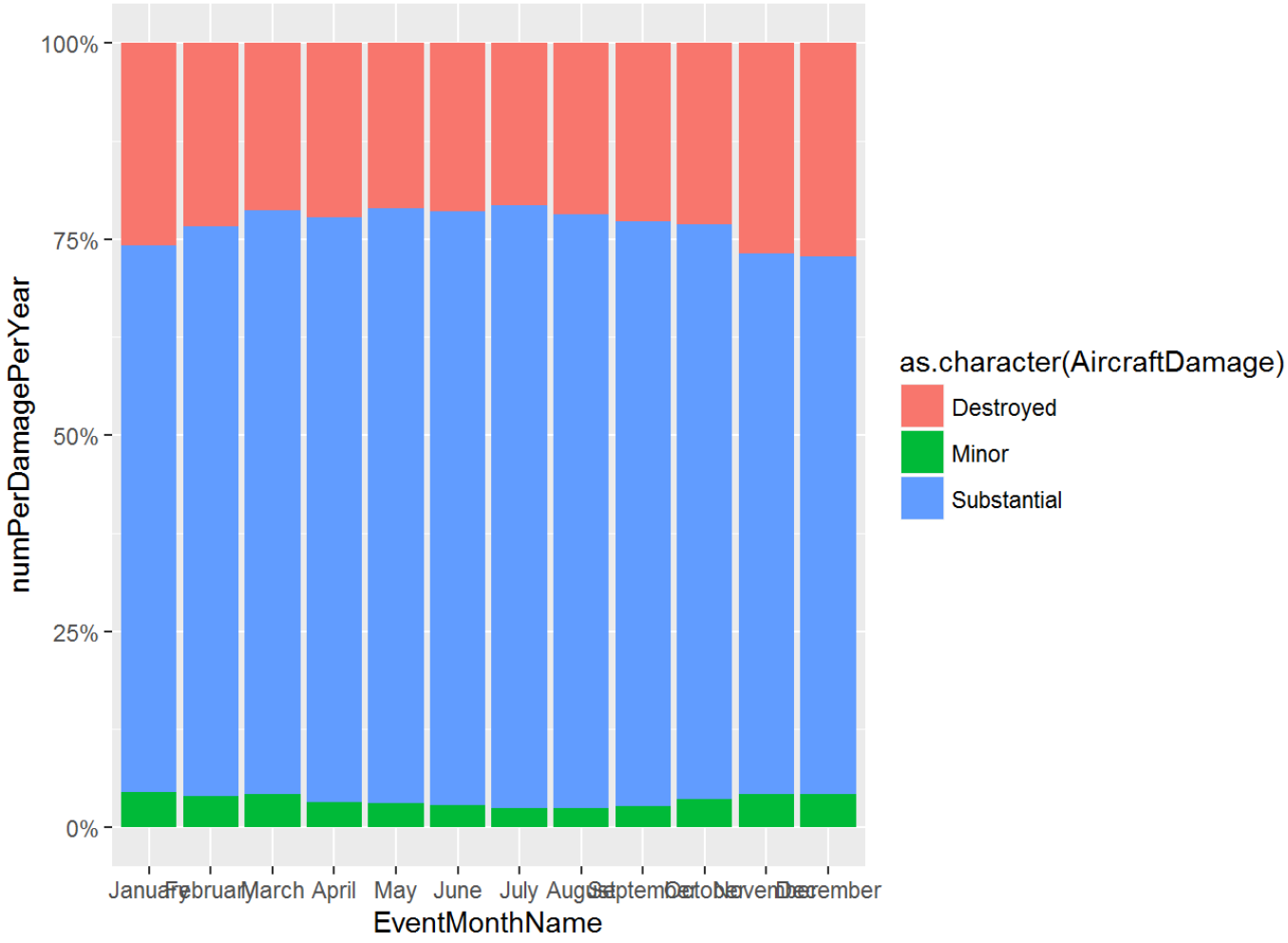
```
events$AircraftDamage <- relevel(events$AircraftDamage, 2)
events$AircraftDamage <- relevel(events$AircraftDamage, 4)
events$AircraftDamage <- relevel(events$AircraftDamage, 4)
levels(events$AircraftDamage)
```

```
## [1] "Minor"       "Substantial" "Destroyed"   "(missing)"
```

```
eventsGrouped2 <- group_by(events, EventMonthName, AircraftDamage) %>%
    filter(AircraftDamage != "(missing)")
# summarize for each year and cluster type
(by_damage <- summarise(eventsGrouped2, numPerDamagePerYear = n()))
```

```
## Source: local data frame [36 x 3]
## Groups: EventMonthName [?]
##
##    EventMonthName AircraftDamage numPerDamagePerYear
##             (fctr)         (fctr)                (int)
## 1         January          Minor                  186
## 2         January    Substantial                 2920
## 3         January      Destroyed                 1084
## 4        February          Minor                  175
## 5        February    Substantial                 3223
## 6        February      Destroyed                 1037
## 7           March          Minor                  238
## 8           March    Substantial                 4218
## 9           March      Destroyed                 1211
## 10          April          Minor                  201
## ..            ...            ...                  ...
```

```
ggplot(by_damage, aes(x = EventMonthName, y = numPerDamagePerYear,
                      fill = as.character(AircraftDamage))) +
    geom_bar(position = "fill", stat = "identity") +
    scale_y_continuous(labels = percent_format())
```

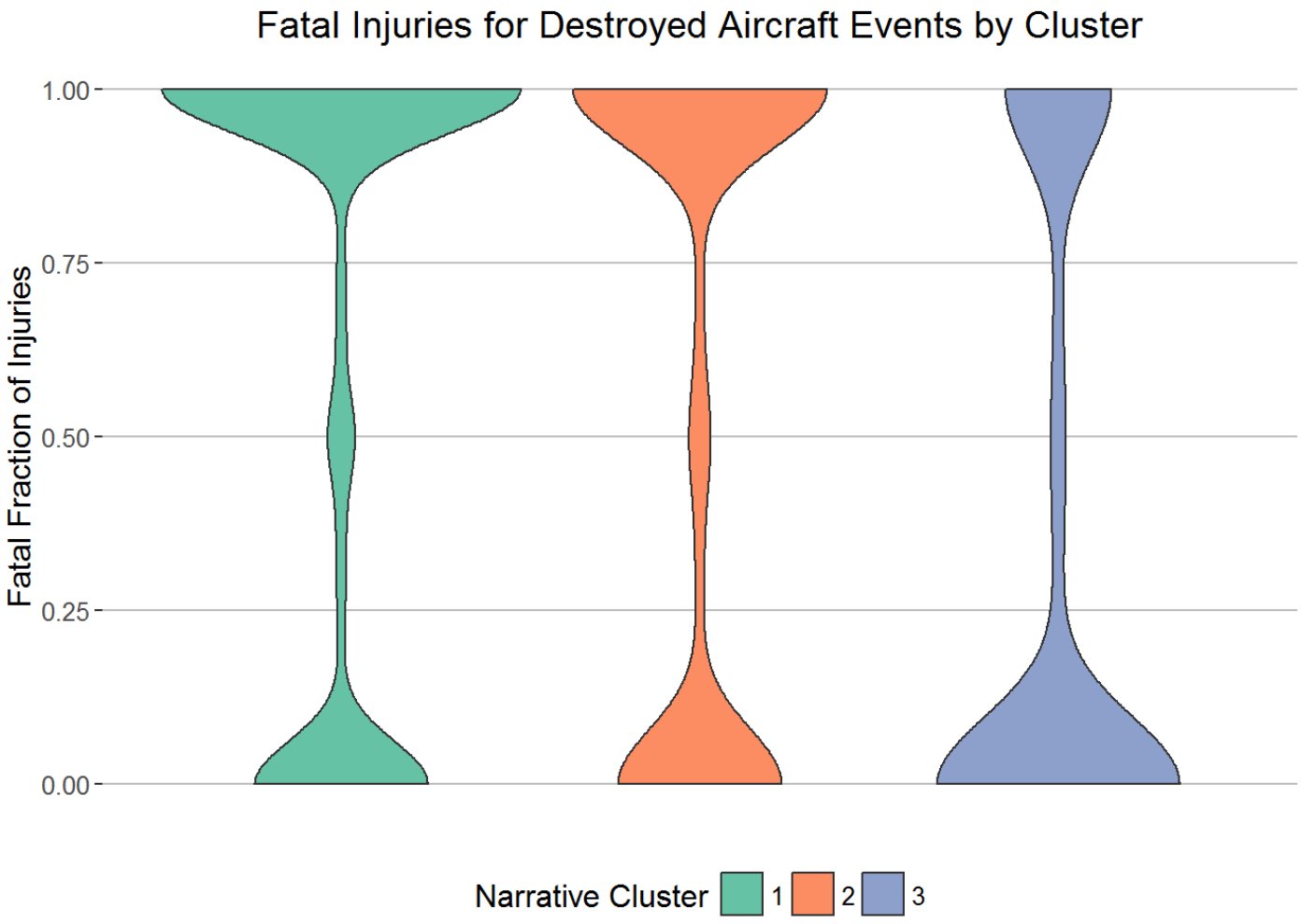


```
    # destroyed aircraft are slightly more likely in the winter
```

Figure D

Examine the relationships between aircraft damage and casualties.

```
ggDfinal <- ggplot(filter(events, AircraftDamage == "Destroyed", !is.na(NarrativeCluste
r)),
                aes(x = AircraftDamage, y = FractionFatalInjuries)) +
    geom_violin(aes(fill = as.character(NarrativeCluster))) +
    scale_x_discrete(breaks = "") +
    theme_hc() +
    theme(legend.margin = unit(0, "cm")) +
    scale_color_brewer(type = "qual", palette = 7, name = "Narrative Cluster") +
    scale_fill_brewer(type = "qual", palette = 7, name = "Narrative Cluster") +
    xlab("") +
    ylab("Fatal Fraction of Injuries") +
    ggtitle("Fatal Injuries for Destroyed Aircraft Events by Cluster")
ggDfinal
```



```
    # For destroyed aircraft, the fraction of fatalities among passengers
    # is highest for narrative cluster 1 and lowest for cluster 3.
```

The following plot shows that fatalities are more likely as aircraft damage intensifies.

```
ggplot(filter(events, AircraftDamage != "(missing)"),
              aes(x = AircraftDamage, y = FractionFatalInjuries)) +
    geom_violin()
```