Analysis Markdown: Study of the Effect of World-Wide Air Terrorism on United States Passenger Airline Performance

Anne L. Sallaska

The following outlines the analysis, which was performed using R. Each section below corresponds to code for the corresponding section in the paper. The `dplyr, lubridate, xtable, gridExtra, ggplot` libraries are used liberally throughout.

Introduction

First, data previously downloaded from the Global Terrorism Database is read into R (http://www.start.umd.edu/gtd/search/). A new column is added to convert the string date into a POSIXct object that can be used in calculations. Columns for number of fatalities and injured are converted from strings to numbers, and a new data frame is formed extracting the domestic terror data and the "shoe bombing" event in France. This will be the base terror data for the analysis. Additional events from around the world will be added later.

```
terrorcsv<-read.csv("GTD-Export_Air.csv",stringsAsFactors=F)
terrorfull<-terrorcsv %>% mutate(d=ymd(DATE))
terror<-terrorfull %>%
        mutate(FATALITIES=as.numeric(FATALITIES),INJURED=as.numeric(INJURED))
terrorUS<-terror %>% filter(COUNTRY=="United States" | GTD.ID=="200112220002")
```

A quick view of types of data included in this data frame, and a sample of their first few values are:

```
str(terrorUS)
```

```
## 'data.frame':    68 obs. of  24 variables:
##  $ GTD.ID       : chr  "201311010046" "201304180010" "200912250024" "200202040010" ...
##  $ DATE         : chr  "2013-11-01" "2013-04-18" "2009-12-25" "2002-07-04" ...
##  $ COUNTRY      : chr  "United States" "United States" "United States" "United States" ...
##  $ CITY         : chr  "Los Angeles" "McCook" "Detroit" "Los Angeles" ...
##  $ PERPETRATOR.1: chr  "Individual" "Unknown" "Al-Qa`ida in the Arabian Peninsula (AQAP)" "Individual
##  $ GUNCERTAIN.1 : int  -9 -9 0 0 0 0 0 0 0 0 ...
##  $ PERPETRATOR.2: chr  "" "" "" "" ...
##  $ GUNCERTAIN.2 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ PERPETRATOR.3: chr  "" "" "" "" ...
##  $ GUNCERTAIN.3 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ FATALITIES   : num  1 0 0 3 0 ...
##  $ INJURED      : num  8 0 2 4 1 0 106 NA NA 0 ...
##  $ TARGET.TYPE.1: chr  "Airports and Aircraft" "Airports and Aircraft" "Airports and Aircraft" "Airpo
##  $ TARGET.TYPE.2: chr  "Government (General)" "" "" "Private Citizens & Property" ...
##  $ TARGET.TYPE.3: chr  "" "" "" "" ...
##  $ REGION       : chr  "North America" "North America" "North America" "North America" ...
##  $ ATTACK.TYPE.1: chr  "Armed Assault" "Bombing/Explosion" "Bombing/Explosion" "Armed Assault" ...
##  $ ATTACK.TYPE.2: chr  "" "" "" "" ...
##  $ ATTACK.TYPE.3: chr  "" "" "" "" ...
```

```
##  $ WEAPON.TYPE.1: chr  "Firearms" "Explosives/Bombs/Dynamite" "Explosives/Bombs/Dynamite" "Firearms"
##  $ WEAPON.TYPE.2: chr  "Firearms" "" "" "Melee" ...
##  $ WEAPON.TYPE.3: chr  "" "" "" "" ...
##  $ WEAPON.TYPE.4: chr  "" "" "" "" ...
##  $ d            : POSIXct, format: "2013-11-01" "2013-04-18" ...
```

This analysis will focus on the dates (in string "DATES" and POSIXct format "d"), the country, and the total number of fatalities, though all variables are carried throughout in the event that additional analysis needs to be performed.

The following produces the dual histograms of Figure 1. First the main histogram is plotted, followed by a calculation of the margins of the inset, plotting the inset itself, and adding text. Custom breaks were used (`sttime, endtime`) in order to extend the x-axis so that it covered all the data. The base histogram binning of "years" without specifying a start and end time left the axis truncated at 2010, since data only exists until 2013.

```
# to make the dual histogram of terrorism dates
par.save<-par() # par.save$mar = 5.1 4.1 4.1 2.1
pdf(file="terror.pdf",width=9,height=6)
par(par.save)

# main histogram
sttime=ymd("1970-01-01");endtime=ymd("2014-01-01");
hist(terror$d,breaks=seq(sttime,endtime,"years"),col="lightblue",axes=F,
        freq=T,xlab="Date",main="World",xlim=c(sttime,endtime+60*60*24*365))
axis.POSIXct(1,at=seq(sttime,endtime+60*60*24*365,"5 years"))
axis(2,col="black")

# calculate position of inset
plotdim <- par("plt")
xleft    = plotdim[2] - (plotdim[2] - plotdim[1]) * 0.35
xright   = plotdim[2]*.98  #
ybottom  = plotdim[4] - (plotdim[4] - plotdim[3]) * 0.38  #
ytop     = plotdim[4]  #
# set position for inset
par(fig = c(xleft, xright, ybottom, ytop), mar=c(0,0,0,0), new=T)

# inset histogram
hist(terrorUS$d,breaks=seq(sttime,endtime,"years"),col="lightblue",axes=F,
        freq=T,xlab="Date",main="",xlim=c(sttime,endtime+60*60*24*365))
axis.POSIXct(1,at=seq(sttime,endtime+60*60*24*365,"5 years"))
axis(2,col="black")
text(x=ymd("1991-06-01"),y=7.8,"U.S.",font=2)
dev.off()
```

The next section forms the complete terror data set used in the analysis, binding the United States terror events with others around the world that had greater than 100 fatalities and occurred after 1987, when the flight data became available. It then arranges the data in decreasing number of fatalities. On-time airline performance data from the United States Department of Transportation's Bureau of Transportation Statistics, previously downloaded from the web (http://www.transtats.bts.gov/tables.asp?db_id=120&DB_Name=Airline%20On-Time%20Performance%20Data). Flight data was retrieved according to the event month and the two bordering months, and each is labeled as "YYYY-MM.csv" in a nested folder called Data. A column, "f", is added corresponding to the flight data file name associated with the month the event occurred, and another column is added which checks if the respective data file exists and is accessible.

```
terrorComp<-filter(rbind(terrorUS,
        filter(terror,FATALITIES>100 & COUNTRY!="United States")),d> ymd("1987-11-01"))
terrorComp<-terrorComp %>% arrange(desc(FATALITIES)) %>%
        mutate(f=paste("./Data/",format(d,format="%Y-%m"),".csv",sep=""))
terrorComp$Exist<-sapply(terrorComp$f,file.exists)
```

September 11, 2001 is a unique case in this data set. Because four flights were hijacked, there are four entries in the Global Terorrism Database. These were grouped together into a single event in the following code because flight data is date based. First, the four entries are subsetted from the main data frame. A single row is created from the first entry, to capture the columns of strings that are the same for each. Then the number of injured and fatalities are summed from each event, and the three separate cities are concatenated. Then these four events are removed from the main data frame, and the new row containing the summaried information is binded to it in their place. Then the data frame is arranged again by number of fatalities.

```
sept11<-filter(terrorComp,DATE=="2001-09-11")
sept11row<-sept11[1,]
sept11row<-sept11row %>% mutate(FATALITIES=sum(sept11$FATALITIES)
        ,INJURED=sum(sept11$INJURED),CITY=paste(unique(sept11$CITY),collapse=", "))
terrorComp<-filter(terrorComp,DATE!="2001-09-11")
terrorComp<-rbind(terrorComp,sept11row)
terrorComp<-arrange(terrorComp,desc(FATALITIES))
```

Using the above data frame, the following selects the pertinent columns as a data frame for Table 1, which lists the terror events selected for this analysis and some of their details. Then the new data frame is printed as a latex table (some formatting changes were applied in latex):

```
# Table 1: summary of terror events for paper
forpaper<-terrorComp
forpaper<-forpaper %>% mutate(CityCountry=paste(CITY,COUNTRY,sep=", "))
forpaper<-select(forpaper,DATE,CityCountry,ATTACK.TYPE.1,WEAPON.TYPE.1,FATALITIES)
print(xtable(forpaper),include.rownames=F)
```

Analysis Overview and Summary Statistics for the Selected Terror Events

Next, the flight data is imported. Each flight has its own entry in a *csv file, which are grouped by month and include statistics such as delay time and if the flight was canceled or not. The following explains some of the variables in the flight data, as well as new summary variables.

```
# Description of flight data columns:
# DEP_DEL15: 0,1, if delayed > 15 min
# DEP_DEL: 0,1 if delayed at all
# DEP_DELAY: total number minutes flight is delayed (early departures count as
# negative minutes)
# DEP_DELAY_NEW: total number minutes flight is delayed (IF flight delayed. Early
# departures set to 0 min.)
# DEP_DEL_min15: total number minutes flight is delayed, if delayed at least 15 minutes

# Summary statistics added, each are aggregated by day:
# mdel,ssdel: mean/sd of mean of minutes flights are delayed per day (including zeros)
```

```
# ndelay,ndelay15: sum of total number of delays per day, at all or by > 15 min
# delamintot,delamintot15: sum of total minutes delayed per day, at all or by > 15 min
# can: sum of cancelled flights per day
# count: total number of flights per day

# if flight is cancelled, all delay stats are NA.
```

Below is a preview of one of the flight data sets (September 2001):

```
## 'data.frame':    490698 obs. of  21 variables:
##  $ FL_DATE              : Factor w/ 30 levels "2001-09-01","2001-09-02",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ UNIQUE_CARRIER       : Factor w/ 12 levels "AA","AQ","AS",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ ORIGIN_AIRPORT_ID    : int  12892 12478 12892 11298 13830 11298 12173 12892 10721 12892 ...
##  $ ORIGIN_CITY_MARKET_ID: int  32575 31703 32575 30194 33830 30194 32134 32575 30721 32575 ...
##  $ ORIGIN_CITY_NAME     : Factor w/ 216 levels "Abilene, TX",..: 122 145 122 49 97 49 87 122 24 122
##  $ ORIGIN_STATE_ABR     : Factor w/ 51 levels "AK","AL","AR",..: 5 33 5 43 10 43 10 5 18 5 ...
##  $ DEST_AIRPORT_ID      : int  12478 12892 12478 12173 11298 13830 11298 12478 12892 10721 ...
##  $ DEST_CITY_MARKET_ID  : int  31703 32575 31703 32134 30194 33830 30194 31703 32575 30721 ...
##  $ DEST_CITY_NAME       : Factor w/ 216 levels "Abilene, TX",..: 145 122 145 87 49 97 49 145 122 24
##  $ DEST_STATE_ABR       : Factor w/ 51 levels "AK","AL","AR",..: 33 5 33 10 43 10 43 33 5 18 ...
##  $ DEP_DELAY            : num  -3 9 -3 6 -8 11 -2 0 -6 -5 ...
##  $ DEP_DELAY_NEW        : num  0 9 0 6 0 11 0 0 0 0 ...
##  $ DEP_DEL15            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ CANCELLED            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ CANCELLATION_CODE    : logi  NA NA NA NA NA NA ...
##  $ CARRIER_DELAY        : logi  NA NA NA NA NA NA ...
##  $ WEATHER_DELAY        : logi  NA NA NA NA NA NA ...
##  $ NAS_DELAY            : logi  NA NA NA NA NA NA ...
##  $ SECURITY_DELAY       : logi  NA NA NA NA NA NA ...
##  $ LATE_AIRCRAFT_DELAY  : logi  NA NA NA NA NA NA ...
##  $ X                    : logi  NA NA NA NA NA NA ...
```

Next, all flight data is read in for each terror event in sets of the three months associated with the event. A summary data frame, `summarydf`, includes the relevant flight data summarized by day. `mindelay` and `tdelay` are vectors that capture the delay time (including 0 for on-time departures, with early departures set to 0) and whether or not the flight was on a terror event date (0,1). The latter has the value of 2 if the event is September 11, 2001. This will be useful in doing the hypothesis tests for statistics in Table 2 later in the analysis. Finally, the summary data frame is merged with the terror data frame. Because of the complexity of the calculation, a loop over each terror event is used instead of some type of `apply` or `mutate` function.

```
summarytot<-data.frame()
mindelay<-vector()
tdelay<-vector()
for (i in 1:nrow(terrorComp)){
        #create combined data frame for flight data on either side of event
        dterror<-terrorComp$d[i]

        # to keep track of progress
        print(c(i,"working on terror date:",format(dterror,format="%Y-%m-%d")))

        # to find the names of the adjacent month data needed to be read in
        t<-strsplit(terrorComp$f[i],"[.(\\/Data)]")[[1]][8]
        t<-paste(t,"-01",sep="")
```

```
        ftemp1<-paste("./Data/",format(ymd(t)-months(1),format="%Y-%m"),".csv",sep="")
        ftemp2<-paste("./Data/",format(ymd(t)+months(1),format="%Y-%m"),".csv",sep="")
        fls<-c(ftemp1,terrorComp$f[i],ftemp2) # file names to be read
        df<-data.frame()
        for (j in 1:3){
                df<-rbind(df,read.csv(fls[j])) # read in each, bind to common frame
        }
        df$d<-ymd(df$FL_DATE)

        # add some binary variables to help with sorting
        df<-df %>% mutate(DEP_DEL=ifelse(DEP_DELAY_NEW>0,1,0)
                ,DEP_DEL_min15=ifelse(DEP_DEL15==1,DEP_DELAY_NEW,0),
                 tdelay=ifelse(d==ymd("2001-09-11"),2,ifelse(dterror==d,1,0)))

        # vectors of delay time (>=0) and if corresponding delay time occurred on a
        # terror date
        mindelay<-c(mindelay,df$DEP_DELAY_NEW)
        tdelay<-c(tdelay,df$tdelay)

        temp<-apply(select(df,d,DEP_DELAY:CANCELLED)[,-1],2,function(z) sum(is.na(z)))
        names(temp)[1:3]<-"temp"

        # run checks to ensure the data is valid
        if (all.equal(temp[1],temp[2],temp[3])==F | temp[4]!=0){
                print("Delay and cancellation NAs not adding up")
        }

        # create summary data frame, aggregating by day
        summarydf<-as.data.frame(summarise(group_by(df,d),count=n(),can=sum(CANCELLED),
                mdel=mean(DEP_DELAY_NEW,na.rm=T),sddel=sd(DEP_DELAY_NEW,na.rm=T),
                ndelay15=sum(DEP_DEL15,na.rm=T),delamintot=sum(DEP_DELAY_NEW,na.rm=T),
                ndelay=sum(DEP_DEL,na.rm=T),delamintot15=sum(DEP_DEL_min15,na.rm=T))

        # reconfigure terror data frame to be bound with summary data frame
        trow<-terrorComp[i,]
        tmerge<-trow[rep(seq_len(nrow(trow)),nrow(summarydf)),]
        tmerge<-tmerge %>% rename(dterror=d,DATE_TERROR=DATE) %>% select(-(f:Exist))

        # combine summary data frame with terror data frame for ease of comparitive
        # analysis
        summarytot<-rbind(summarytot,cbind(summarydf,tmerge))
}
```

Then some additional parameters are added to the summary data frame to help with analysis and plotting, including the daily cancellation rate, `canrate`:

```
summarytot<-summarytot %>% mutate(USword=as.factor(paste(COUNTRY,FATALITIES,sep=": ")),
        terror=ifelse(dterror==d,T,F),
        canrate=can/count*100,
        weekday=ifelse(wday(d)==1 | wday(d)==7,0,1))
```

Next, some summary statistics are calculated:

```
flightstats<-select(summarytot,d,terror,count,can,ndelay15:delamintot15)
apply(flightstats[,-1],2,sum,na.rm=T) # total sums of each column
apply(filter(flightstats[,-1],terror==F),2,sum,na.rm=T) # total on days without terror
apply(filter(flightstats[,-1],terror==T),2,sum,na.rm=T) # total on days with terror
```

The data frame `flightstats` and the vector `mindelay` are then read into a function `statsfun` in order to produce the statistics in Table 2. `mindelay` is subsetted to pick out statistics for varying delay criteria. The results are then bound in a data frame and returned. The function is:

```
statsfun<-function(flightstats,mindelay){
        flightstatsdf<-as.data.frame(summarise(flightstats,n=sum(count),ndays=n(),
               Scheduled.flights.per.day=mean(count),
               Scheduled.flights.per.day.sd=sd(count),
               ncancel=sum(can),nreal=n-ncancel,ndelay=sum(ndelay),ndelay15=sum(ndelay15),
               Rate.cancelled=(ncancel/n*100),
               Rate.delayed=(ndelay/nreal*100),
               Rate.delayed.by.gt.15.min=(ndelay15/nreal*100)))
        mindelay<-mindelay[!is.na(mindelay)]
        mindelaydelay<-mindelay[mindelay>0]
        mindelaydelay15<-mindelay[mindelay>15]
        minstats<-data.frame(Average.delay=mean(mindelay),Average.delay.sd=sd(mindelay),
             Average.delay.if.delayed=mean(mindelaydelay),
             Average.delay.if.delayed.sd=sd(mindelaydelay),
             Average.delay.if.delayed.gt.15.min=mean(mindelaydelay15),
             Average.delay.if.delayed.gt.15.sd=sd(mindelaydelay15))
        flightstatsdf<-cbind(flightstatsdf,minstats)

        return(flightstatsdf)
}
```

Below is the code to call the above function for the 5 subsets of data in Table 2, for varying groupings of terror and non-terror events. Then the data frames from each function call are bound and printed as a latex table, which is the basis for Table 2 (with some formatting changes):

```
# get summary statistics from user-defined function
flightstatsdf1<-statsfun(flightstats,mindelay) # all dates
# non-terror event dates:
flightstatsdf2<-statsfun(filter(flightstats,terror==F),mindelay[tdelay==0])
# terror-event dates:
flightstatsdf3<-statsfun(filter(flightstats,terror==T),mindelay[tdelay!=0])
flightstatsdf4<-statsfun(filter(flightstats,terror==T & d!=ymd("2001-09-11")),
        mindelay[tdelay==1]) # terror-event dates not 9-11
flightstatsdf5<-statsfun(filter(flightstats,terror==T & d==ymd("2001-09-11")),
        mindelay[tdelay==2]) # terror-event 9-11

flightstatsdf<-rbind(flightstatsdf1,flightstatsdf2,flightstatsdf3,
        flightstatsdf4,flightstatsdf5)

# Table 2
print(xtable(t(select(flightstatsdf,-(ncancel:ndelay15)))))
```

Distribution of Scheduled Number of Flights

The confidence interval for the rate of flights scheduled per day is calculated as:

```
ndays<-nrow(summarytot)
m<-mean(summarytot$count);s<-sd(summarytot$count)
m+c(-1,1)*qnorm(.975)*s/sqrt(ndays) # using normal distribution
m+c(-1,1)*qt(.975,ndays-1)*s/sqrt(ndays) # using t-distribution
```

The following is code to produce Figure 2. First a new data frame is constructed from the `summarytot` data frame, which contains all aggregated flight data per day. Then the events surrounding September 11 are cut into three separate sections in order to observe the sharp changes in scheduled number of flights.

```
xbox<-summarytot %>% select(dterror,d,count) # subset only relevant columns

# this returns a logical vector if the date criteria are met
ind<-xbox$dterror==ymd("2001-09-11") & xbox$d>=ymd("2001-09-29")

# change associated terror date so this data will
# be grouped separately from September 11 on the plot
xbox$dterror[ind]<-ymd("2001-09-29")  # group 2
ind<-xbox$dterror==ymd("2001-09-11") & xbox$d<ymd("2001-09-11") # for group 3
xbox$dterror[ind]<-ymd("2001-08-01") # group 3

# Add Decade column in order to color the data based on decade.
xbox<-xbox %>% mutate(Decade=as.factor(ifelse(dterror<ymd("1990-01-01"),"1980s",
        ifelse(dterror<ymd("2000-01-01"),"1990s",
        ifelse(dterror<ymd("2010-01-01"),"2000s","2010s")))))

# change level order so that the '80s are on the bottom of the plot legend
xbox$Decade<-factor(xbox$Decade, levels=rev(levels(xbox$Decade)))

cols<-brewer.pal(7,"PuRd")
pal<-colorRampPalette(cols)
pdf(file="DistFlights.pdf",width=13,height=6)
gbox<-ggplot(xbox,aes(x=factor(dterror),count))
gbox<-gbox+theme_bw()+coord_flip()
gbox<-gbox+labs(x="Event Date",y="Rate of Scheduled Flights Per Day")
gbox<-gbox+geom_boxplot(aes(fill=Decade))
gbox<-gbox+scale_fill_manual(values=pal(4))
gbox<-gbox+theme(axis.title.x=element_text(vjust=-0.25),
        axis.title.y=element_text(vjust=1.3))
dev.off()
```

Distribution of Delays

First, the September 2001 data is read in separately:

```
# isolate 9/11 data
x911<-read.csv("./Data/2001-09.csv")
x911$d<-ymd(x911$FL_DATE)
x911<-x911 %>% mutate(d911=ifelse(d==ymd("2001-09-11"),1,0),
        weekday=ifelse(wday(d)==1 | wday(d)==7,0,1))
```

Testing median compared to means:

```
# to test average delay times above zero, mean vs median
q<-quantile(filter(x911,DEP_DELAY_NEW > 0)$DEP_DELAY_NEW,probs=c(0.16,.84))
summary(filter(x911,DEP_DELAY_NEW > 0)$DEP_DELAY_NEW)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    4.00   11.00   27.12   31.00 1477.00
```

```
mdn<-summary(filter(x911,DEP_DELAY_NEW > 0)$DEP_DELAY_NEW)[3]
c(1,-1)*mdn-q*c(1,-1)
```

```
## 16% 84%
##   8  38
```

```
mean(filter(x911,DEP_DELAY_NEW > 0)$DEP_DELAY_NEW);sd(filter(x911,
        DEP_DELAY_NEW > 0)$DEP_DELAY_NEW)
```

```
## [1] 27.11746
```

```
## [1] 43.93077
```

To generate histograms of individual flight delays for each date in September 2001 for Figure 3:

```
pdf(file="Delays.pdf",width=11)
my_breaks<-c(1,10,100,500,1500) # for custom graident coloring
g<-ggplot(filter(x911,DEP_DELAY_NEW > 0),aes(x=DEP_DELAY_NEW/60, fill=..count..))
g<-g+geom_histogram(color="black")
g<-g+theme_bw()
g<-g+scale_fill_gradient("Frequency",low = "springgreen",
        high = "darkgreen",breaks=my_breaks,labels=my_breaks,trans="log")
g<-g+facet_wrap(~d,scales="free_x")
g<-g+scale_y_log10()
g<-g+labs(x="Delay (hours)", y="Frequency")
g
dev.off()
```

Significance of Differences in Parameters

The hypothesis tests to investigate the significance of the differences in terror event statistics compared to non-terror event statistics are up next, using the functions `ratehyp` for the rates and `avghyp` for the average delay times:

```
ratehyp<-function(flightstatsdf,i,j){
        # find the rates for non-terror and terror subsets, based on input
        # variables i and j
        r_noterror<-select(flightstatsdf[i,],Rate.cancelled:Rate.delayed.by.gt.15.min)/100
        r_terror<-select(flightstatsdf[j,],Rate.cancelled:Rate.delayed.by.gt.15.min)/100
        n_terror<-flightstatsdf$n[j] # number of terror events in the data set
        s<-sqrt(r_noterror*(1-r_noterror)/n_terror) # estimated variance
```

```
        # test statistic to compare to 0.05, negative sign just to be able to use the pt
        # function and not introduce lower.tail=T
        tstat<--abs(r_noterror-r_terror)/s

        # calculate the probability associated with the test statistic for
        # n_terror-1 degrees of freedom
        pval<-sapply(tstat,function(z) 2*pt(z,n_terror-1))
     return(pval)
}

avghyp<-function(m_noterror,m_terror){
        m_noterror<-m_noterror[!is.na(m_noterror)] # remove NAs from non-terror events

        # make a list of different subsets of delay data based on delay time
        # for non-terror events
        m<-list(m_noterror,
                m_noterror[m_noterror>0],m_noterror[m_noterror>15] )

        m_terror<-m_terror[!is.na(m_terror)] # remove NAs from terror events

        # make a list of different subsets of delay data based on delay time
        # for terror events
        mt<-list(m_terror,
                m_terror[m_terror>0],m_terror[m_terror>15] )

        # apply the t.test function
        pval<-sapply(1:3,function(z) t.test(m[[z]],mt[[z]])$p.val)
        return(pval)
}
```

These functions are then called by:

```
# comparing each to non-terror
pvalrate<-sapply(seq(3,5,1),function(i) ratehyp(flightstatsdf,2,i))

# hypothesis test for minutes delayed, unequal variances; non-terror vs terror
pvalavg1<-avghyp(mindelay[tdelay==0],mindelay[tdelay!=0])
# hypothesis test for minutes delayed, unequal variances; non-terror vs terror, non-9/11
pvalavg2<-avghyp(mindelay[tdelay==0],mindelay[tdelay==1])
# hypothesis test for minutes delayed, unequal variances; non-terror vs terror, 9/11
pvalavg3<-avghyp(mindelay[tdelay==0],mindelay[tdelay==2])
pvalavg<-cbind(pvalavg1,pvalavg2,pvalavg3)
rownames(pvalavg)<-c("Avg","Avg if delayed","Avg if delayed > 15")
```

Returning to Normal

Then we move to the hypothesis testing of Section 3 for the cancellation rates and when they return to normal, if there is a deviation after the terror event. The results are then appended to the data frame for terror data, `terrorComp`. Again, because of the complexity, a loop over each terror event is used.

```
terrorComp$mdayback<-0.
terrorComp$sddayback<-0.
for (i in 1:nrow(terrorComp)){
        dterror<-terrorComp$d[i]

        # find the days in the flight summary data frame that are associated with this
        # terror event and subset those out into "subtot"
        ind<-summarytot$dterror==terrorComp$d[i]
        subtot<-summarytot[ind,]

        # create a new variable that indicates whether or not the data is before or after
        # the terror event
        subtot<-subtot %>% mutate(postterror=ifelse(d<dterror,F,T))

        # durations in days for the "previous" data compared to post-event data
        # in the hypothesis test
        tprev<-c(30,15,5)

        # create a list of data frames, each data frame corresponding to the pre-event
        # set of data defined by the number of days of "tprev"
        prev<-lapply(tprev, function(z) filter(subtot,d>=dterror-days(z) & d<dterror))

        # this is the meat of the calculation.  the sapply function takes data for
        # **each day** and runs a t-test on it using the respective pre-event data
        # set and saves the p-value by binding it to the data frame.
        subtot$p.val30<-sapply(subtot$canrate,function(z) t.test(prev[[1]]$canrate-z)$p.val)
        subtot$p.val15<-sapply(subtot$canrate,function(z) t.test(prev[[2]]$canrate-z)$p.val)
        subtot$p.val5<-sapply(subtot$canrate,function(z) t.test(prev[[3]]$canrate-z)$p.val)

        # this creates new logical variables indicating whether the data is pre-event
        # or post-eventf or ease of analysis
        subtot$prev30<-sapply(subtot$d,function(z) z>=prev[[1]]$d[1] &
                z<= prev[[1]]$d[nrow(prev[[1]])])
        subtot$prev15<-sapply(subtot$d,function(z) z>=prev[[2]]$d[1] &
                z<= prev[[2]]$d[nrow(prev[[2]])])
        subtot$prev5<-sapply(subtot$d,function(z) z>=prev[[3]]$d[1] &
                z<= prev[[3]]$d[nrow(prev[[3]])])

        alpha=0.05 # the significance level for 95 % confidence

        # this is the end of the hypothesis test, where the p-value for each day and
        # set of pre-event data is compared to alpha.  The logical value T or F:
        # p-val > alpha => T, the null hypothesis is rejected and the terror event is
        # still disrupting the cancellation  rate; p-value < alpha -> F, the null
        # hypothesis cannot be rejected any longer and rates have returned to normal.
        # This value, T/F, is then bound to the data frame.
        subtot<-subtot %>% mutate(alpha30=ifelse(p.val30>alpha,T,F),
                alpha15=ifelse(p.val15>alpha,T,F),alpha5=ifelse(p.val5>alpha,T,F))

        # save September 11, 2001 data
        if (i==1){
             sept11subtot<-subtot
             sept11prev<-prev
```

```
        }

        # find the point at which the hypothesis test result flips from F to T for
        # each set of pre-event data.  This is the date at which the cancellation  rate
        # returns to normal, according to the test.
        daybackind<-c(which(subtot$alpha30==T & subtot$postterror==T)[1],
                which(subtot$alpha15==T & subtot$postterror==T)[1],
                which(subtot$alpha5==T & subtot$postterror==T)[1])

        # find the difference in the dates above from the terror date
        dayback<-difftime(subtot$d[daybackind],dterror,units="days")

        # compute the average for all three pre-event sets and take the mean and standard
        # devation
        terrorComp$mdayback[i]<-mean(dayback)
        terrorComp$sddayback[i]<-sd(dayback)
}
```

Some summary statistics are calculated for September 11. This gives values like the minimum, maximum, median, mean, and first and third quartiles.

```
# summary of pre-sept 11 scheduled flight stats
summary(filter(sept11subtot,d<ymd("2001-09-10"))$count)
# summary of sept 11 scheduled flight stats
summary(sept11subtot$count)
```

Then finally, a hypothesis test is run to test the significance of the time durations to return to normal, in a similar fashion to the above calculation, though much simpler.

```
n<-3 # number of hypotheses tested with 3 pre-event data sets
terrorComp<-terrorComp %>%
        mutate(pval=ifelse(mdayback>0, # only for events with non-zero times to return
        # calculate the p-value from the test-statistic
        pt(mdayback/(sddayback/sqrt(n)),n-1,lower.tail=F),1),
        reject=ifelse(pval<0.05,T,F)) # reject if p-val < 0.05

# view the results for the specified columns
select(terrorComp,d,COUNTRY, mdayback,sddayback,pval,reject)
```

The results of the above analysis, including the time to return to normal for each terror event, are selected and converted into a latex table, which is the basis for Table 3 (plus formatting changes):

```
# Table 3
forpaper1<-terrorComp
forpaper1<-forpaper1 %>% mutate(CityCountry=paste(CITY,COUNTRY,sep=", "))
forpaper1<-select(forpaper1,DATE,COUNTRY,FATALITIES,mdayback,sddayback,reject)
print(xtable(forpaper1),include.rownames=F)
```

Plots for Section 3

Code for Figure 4:

```
pdf(file="cancellation _rates.pdf",width=11)
gcanr<-ggplot(summarytot,aes(x=as.Date(d),y=canrate))+geom_point(size=4,color="salmon")
gcanr<-gcanr+geom_point(size=5,color="black",alpha=0.3)
gcanr<-gcanr+theme_bw()+labs(x="Date",y="Rate of cancellation s per Day")
gcanr<-gcanr+facet_wrap(USword~dterror,scales="free")+scale_x_date()
gcanr<-gcanr+geom_vline(data=summarytot,aes(xintercept=as.numeric(as.Date(dterror))),
        linetype="dashed")
gcanr<-gcanr+theme(axis.text.x=element_text(angle=45,hjust=1),
        axis.title.y=element_text(vjust=1.3))
dev.off()
```

For Figure 5, this was done as three separate plots (`g911a, g911b, g911c`), sewn together for one plot in the paper via the `grid.arrange()` function. A function, `pvalplot`, was written to evaluate the plots, given a data frame, in order to generalize it.

```
pvalplot<-function(sept11subtot){
        sept11subtot$banner<-as.factor("Pre-event data: 30 days")
        g911<-ggplot(filter(sept11subtot,prev30!=T),aes(x=d,y=p.val30))
        g911<-g911+geom_point(size=5)
        g911<-g911+geom_point(data=filter(sept11subtot,prev30==T),aes(x=d,y=p.val30),size=6,col="deepsky
        g911<-g911+geom_hline(yintercept=0.05,linetype="dashed",col="red")
        g911<-g911+labs(x="Date",y="P-Value")
        g911<-g911+facet_grid(~banner)
        g911<-g911+ scale_y_log10() +theme_bw()
        #g911<-g911+annotate("text",x=ymd("2001-08-15"),y=1e-55,label="Pre-event data: 30 days")
        g911<-g911+theme(axis.title.x=element_text(angle=0,vjust=0.1))
        g911
        g911a<-g911
        sept11subtot$banner<-as.factor("Pre-event data: 15 days")
        g911<-ggplot(filter(sept11subtot,prev15!=T),aes(x=d,y=p.val15))
        g911<-g911+geom_point(size=5)
        g911<-g911+geom_point(data=filter(sept11subtot,prev15==T),aes(x=d,y=p.val15),size=6,col="deepsky
        g911<-g911+geom_hline(yintercept=0.05,linetype="dashed",col="red")
        g911<-g911+labs(x="Date",y="P-Value")
        g911<-g911+facet_grid(~banner)
        g911<-g911+ scale_y_log10() +theme_bw()
        #g911<-g911+annotate("text",x=ymd("2001-08-15"),y=1e-25,label="Pre-event data: 15 days")
        g911<-g911+theme(axis.title.x=element_text(angle=0,vjust=0.1))
        g911
        g911b<-g911
        sept11subtot$banner<-as.factor("Pre-event data: 5 days")
        g911<-ggplot(filter(sept11subtot,prev5!=T),aes(x=d,y=p.val5))
        g911<-g911+geom_point(size=5)
        g911<-g911+geom_point(data=filter(sept11subtot,prev5==T),aes(x=d,y=p.val5),size=6,col="deepskybl
        g911<-g911+geom_hline(yintercept=0.05,linetype="dashed",col="red")
        g911<-g911+labs(x="Date",y="P-Value")
        g911<-g911+facet_grid(~banner)
        g911<-g911+ scale_y_log10() +theme_bw()
        #g911<-g911+annotate("text",x=ymd("2001-08-15"),y=1e-7,label="Pre-event data: 5 days")
        g911<-g911+theme(axis.title.x=element_text(angle=0,vjust=0.1))
        g911
        g911c<-g911
        return(list(g911a,g911b,g911c))
```

```
}

gpvalplot<-pvalplot(sept11subtot)
pdf(file="P-values.pdf",width=7,height=10)
grid.arrange(gpvalplot[[1]],gpvalplot[[2]],gpvalplot[[3]],nrow=3)
dev.off()
```

For Figure 6, similar to Figure 5, two separate plots are produced (`ghyp, ghypf`) and are then combined with `grid.arrange()`. Because the fatalities of Colombia and Great Britain overlap on this plot, specific labels were created to separate them.

```
cols<-brewer.pal(3,"Set1")
pal<-colorRampPalette(cols)
laboffset<-c(filter(terrorComp,COUNTRY=="Great Britain")$d,
        filter(terrorComp,COUNTRY=="Colombia")$d)
laboffsetlabels<-c(filter(terrorComp,COUNTRY=="Great Britain")$FATALITIES,
        filter(terrorComp,COUNTRY=="Colombia")$FATALITIES)
ghyp<-ggplot(terrorComp,aes(x=d,y=mdayback,col=COUNTRY,size=log10(FATALITIES+1)))
ghyp<-ghyp+geom_point()
ghyp<-ghyp+geom_hline(yintercept=0,linetype="dashed")
ghyp<-ghyp+geom_errorbar(size=0.4,aes(ymin=mdayback-sddayback,
        ymax=mdayback+sddayback,width=30000000,col=COUNTRY))
ghyp<-ghyp+scale_colour_manual(values=pal(6))+guides(shape=F,color=F,size=F)
ghyp<-ghyp+scale_size(range=c(6,20))
ghyp<-ghyp+theme_bw()+labs(x="Event Date",y="Time to Return to Pre-Terror
        Cancellation Rates (days)")
ghyp<-ghyp+theme(axis.title.x=element_text(angle=0,vjust=0.1),
        axis.title.y=element_text(vjust=1.3))
ghyp<-ghyp+geom_text(size=4,col="white",aes(x=d,
        y=ifelse(COUNTRY!="Great Britain" & COUNTRY!="Colombia",mdayback,mdayback+3)
        ,label=FATALITIES))
ghyp<-ghyp+annotate("text",col="white",size=4,
        x=laboffset+c(-150,100)*3600*24,y=0,label=laboffsetlabels)

ghypf<-ggplot(terrorComp,aes(x=(FATALITIES),y=mdayback))
ghypf<-ghypf+geom_point(aes(col=COUNTRY,shape=COUNTRY),size=5)
ghypf<-ghypf+geom_hline(yintercept=0,linetype="dashed")
ghypf<-ghypf+geom_errorbar(aes(ymin=mdayback-sddayback,ymax=mdayback+sddayback,
        width=.2,col=COUNTRY))
ghypf<-ghypf+scale_colour_manual(values=pal(6))
ghypf<-ghypf+ scale_x_log10(limits=c(0.1,10000))
ghypf<-ghypf+theme_bw()+labs(x="Number of Fatalities",y="")
ghypf<-ghypf+theme(axis.title.x=element_text(angle=0,vjust=0.1),
        axis.title.y=element_text(vjust=1.3))

pdf(file="Hyp.pdf",width=12,height=6)
grid.arrange(ghyp,ghypf,nrow=1,widths=c(1.2,1.2))
dev.off()
```