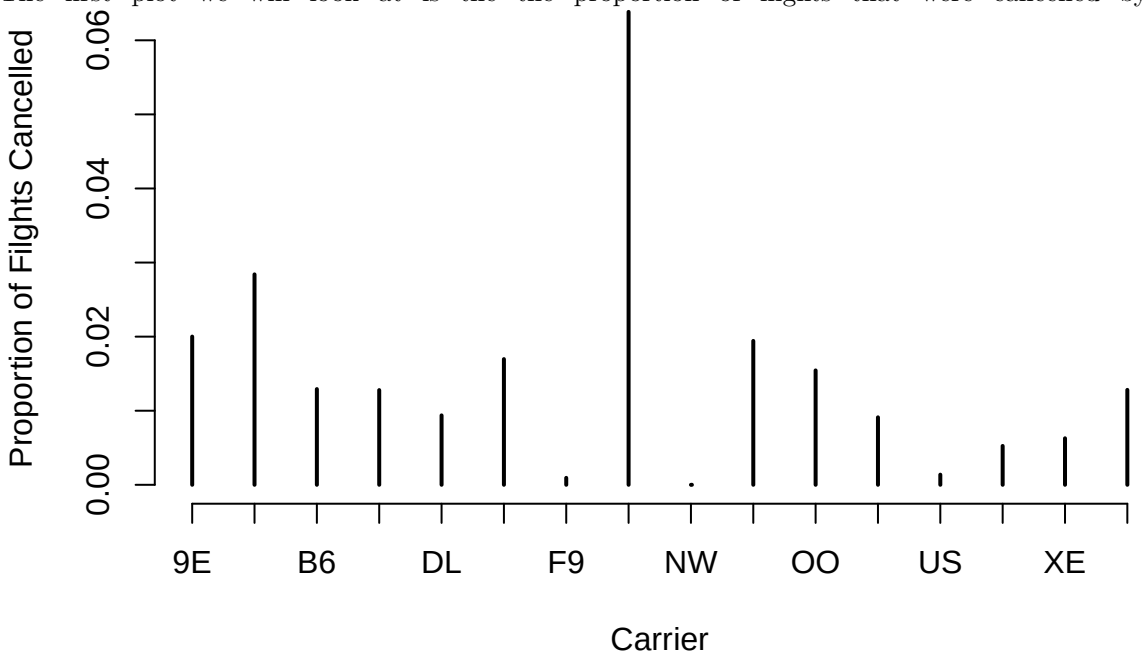
# Exercises 2

*Carlton Washburn*

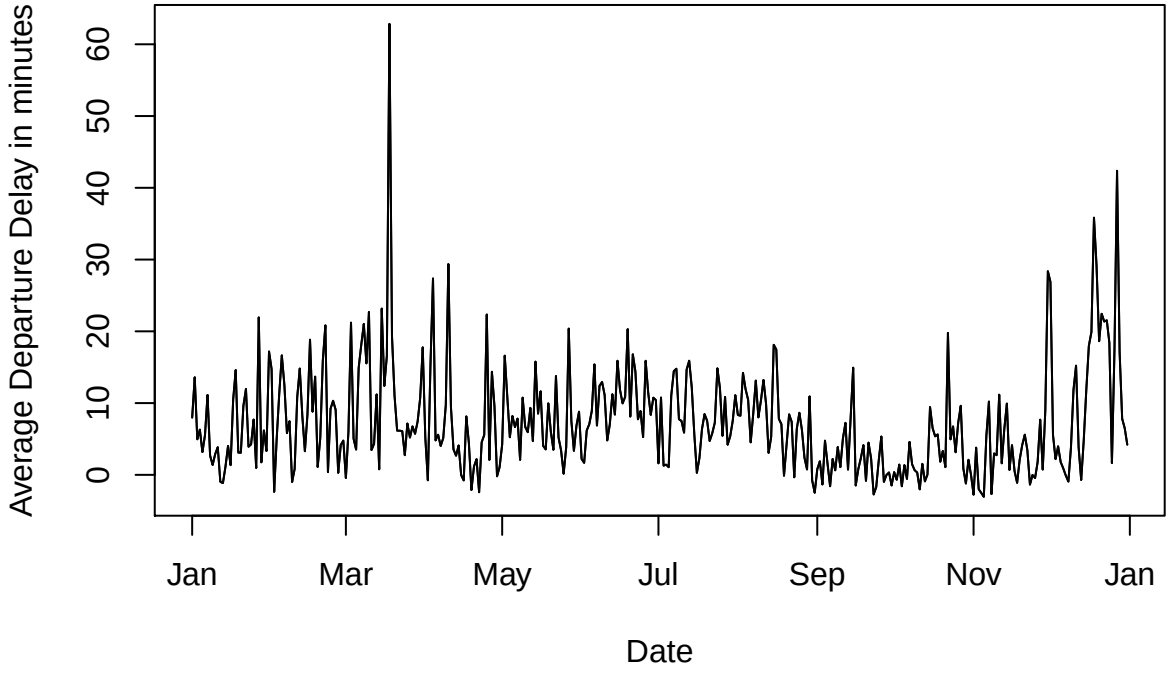
*August 15, 2015*

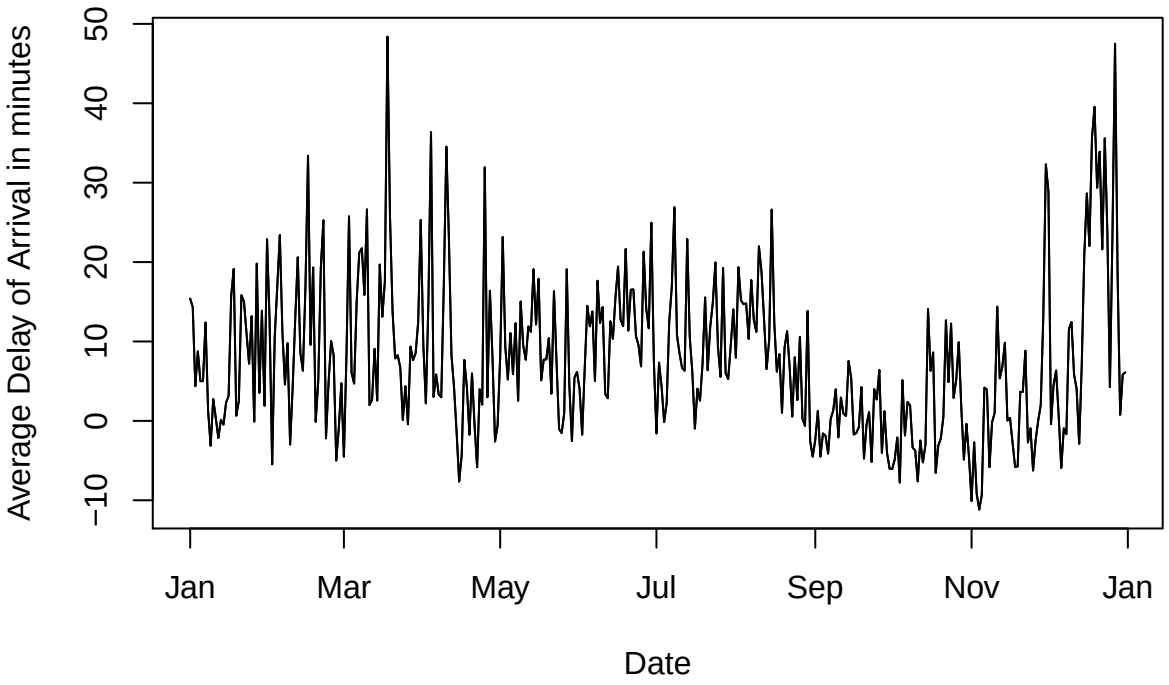
### Flights at ABIA

The first plot we will look at is the the proportion of flights that were cancelled by carrier.




The next two plots we will look at are the Average delay in arrival or departure for every day of the year.






### Author Attribution

In the following section, we try to predict the author of an article based on the words in the article. First we attempt Naive-Bayes.

```
library(tm)
```

```
## Loading required package: NLP
```

```
readerPlain = function(fname){
  readPlain(elem=list(content=readLines(fname)),
            id=fname, language='en') }
author_train = Sys.glob('./ReutersC50/C50train/*')
author_test = Sys.glob('./ReutersC50/C50tests/*')
author_dirs = c(author_train,author_test)
file_list = NULL
labels = NULL
for(author in author_dirs) {
  author_name = substring(author, first=23)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  labels = append(labels, rep(author_name, length(files_to_add)))
}
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

my_corpus = Corpus(VectorSource(all_docs))
names(my_corpus) = file_list
my_corpus = tm_map(my_corpus, content_transformer(tolower)) # make everything lowercase
```

```r
my_corpus = tm_map(my_corpus, content_transformer(removeNumbers)) # remove numbers
my_corpus = tm_map(my_corpus, content_transformer(removePunctuation)) # remove punctuation
my_corpus = tm_map(my_corpus, content_transformer(stripWhitespace)) ## remove excess white-space
my_corpus = tm_map(my_corpus, content_transformer(removeWords), stopwords("SMART"))

DTM = DocumentTermMatrix(my_corpus)
DTM = removeSparseTerms(DTM, 0.975)
X = as.matrix(DTM)
authors = levels(as.factor(labels))
i = 0;
w = NULL
smooth_count = 1/nrow(X)
for (author in authors) {
  col = NULL
  train = X[((i*50)+1):((i*50)+50),]
  col = colSums(train + smooth_count)
  names(col) = colnames(train)
  temp = col/sum(col)
  w = cbind(w,temp)
  i = i+1
}
colnames(w) = authors

fit = matrix(nrow = 2500, ncol = 50)
test = X[2501:5000,]
for (i in seq(1:2500)){
  for ( j in seq(1:50)){
    fit[i,j] = sum(test[i,]*log(w[,j]))
  }
}
colnames(fit) = authors
labels_fit = apply(fit,1,which.max)
author_fit = authors[labels_fit]

good = 0
for (i in seq(1, 2500)){
  if (author_fit[i] == labels[2500+i]){
    good = good + 1
  }
}
good/2500
```

```
## [1] 0.5228
```