Old Faithful

J.A.Hartigan Yale University
September 2013

1 Old Faithful History

from http://en.wikipedia.org/wiki/Old_Faithful

Old Faithful is a cone geyser located in Wyoming, in Yellowstone National Park in the United States. Old Faithful was named in 1870 during the Washburn-Langford-Doane Expedition and was the first geyser in the park to receive a name. Eruptions shoot 14,000 to 32,000 litres of boiling water to a height of 32 to 56 metres lasting from 1.5 to 5 minutes. Intervals between eruptions can range from 45 to 125 minutes, averaging 66.5 minutes in 1939, slowly increasing to an average of 90 minutes apart today. Harry Woodward (1938) first suggested a linear relationship between the duration of the present eruption, and the interval to the next eruption, which the rangers used to predict the time of the next eruption for visitors. The lengthening intervals in recent years made the regression model increasingly inaccurate, and a simpler scheme is used: Old Faithful will erupt 65 minutes after an eruption lasting less than 2.5 minutes, or 91 minutes after an eruption lasting more than 2.5 minutes. The reliability of Old Faithful can be attributed to the fact that it is not connected to any other thermal features of the Upper Geyser Basin.
We will use recent Old Faithful durations and intervals to develop a prediction model for the time to the next eruption.

2 The Data

There are two main sources of data:

(1) Temperature based intervals:
From
http://www.geyserstudy.org/geyser.aspx?pGeyserNo=OLDFAITHFUL
These data consists of eruption times in 2000-2011 based on water temperatures measured continuously 20 minutes west of the vent. The water temperature data was collected by Ralph Taylor, and by other personnel working for the Geology Department of the Yellowstone Center for Resources, National Park Service. The eruption times were computed from the temperature readings by Ralph Taylor. The eruption intervals are the differences in eruption times.

(2) Ranger Log transcriptions:
From http://www.geyserstudy.org/ofvclogs.aspx
These data are transcribed by Lynn Stephens, Marion Powell, and Mary Schwarz, from the ranger logs of Old Faithful eruption intervals and durations, from 1970 to 2010. The logs are usually recording during the working day, and in the non-winter months. They are less accurate than Taylor's temperature based intervals, but they have a better length of record, and also have the duration data necessary to evaluate the Woodward model and the present ranger prediction model.

Both sources of data are used in the data preparation session, with the data actually used in the R analysis being eventually saved in "data/Taylor.csv" and "data/Stephens.csv"

Start here in the script from now on

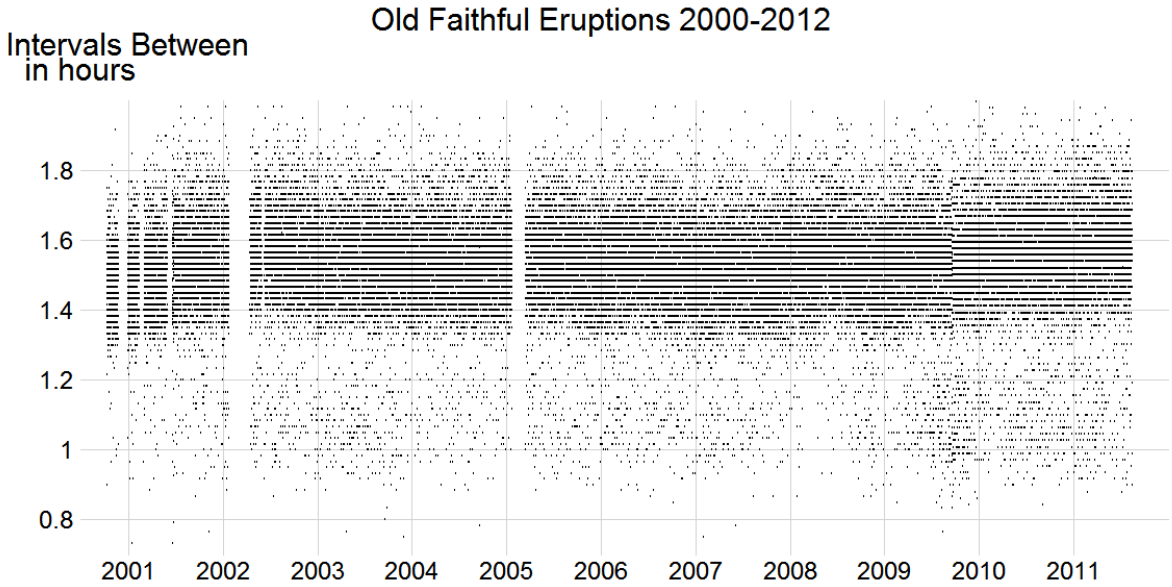
```
oft <- read.csv("data/Taylor.csv", as.is=T, header=T)
options(digits=10)
head(oft,4)
```

```
        time    interval
1 2000.777696 1.350000000
2 2000.777854 1.383333333
3 2000.778023 1.483333333
4 2000.778176 1.350000000
```

3 Old Faithful Intervals 2000-2012

3.1 A scatter plot

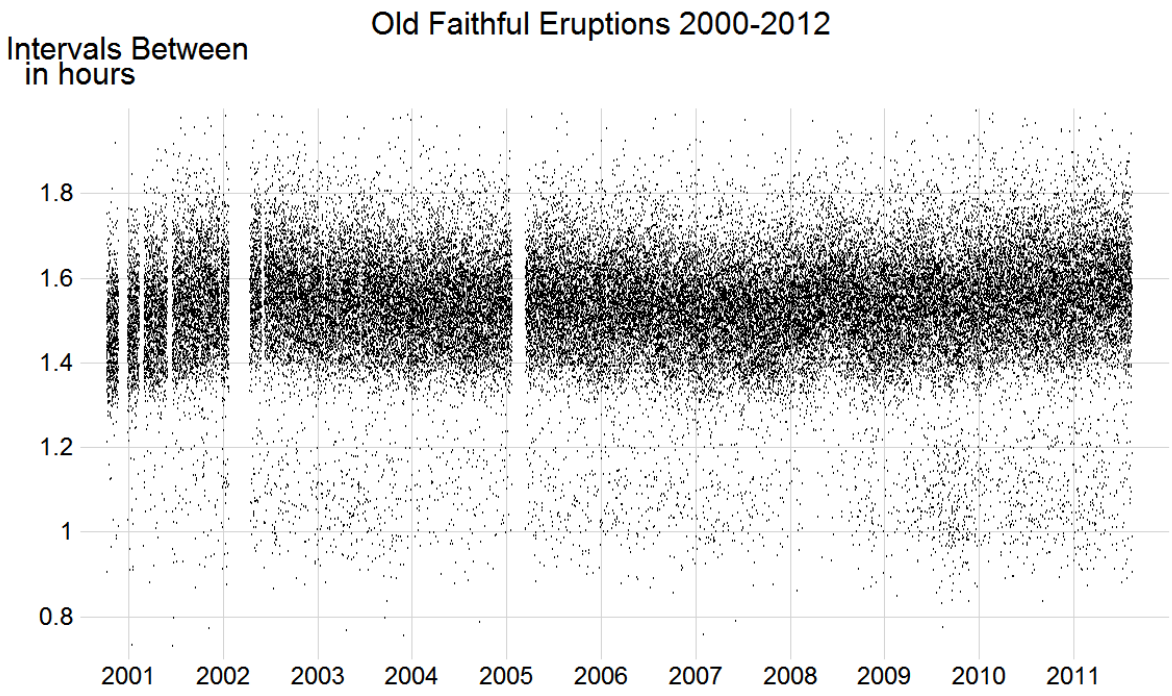
```
tiff("pictures/Taylor Intervals.tif", w=1200, h=600)
use <- oft$interval < 2
Grid(xticks=c(2000.5,2001:2012),
yticks=c(0.7,seq(0.8, 2.0, 0.2)),
ylab = "Old Faithful Eruptions 2000-2012/Intervals
Between/in hours", at=c(2006, 2001, 2000.5), cex=2.5)

points(oft$time[use], oft$interval[use], cex=0.2)
dev.off()
```



Note the dark horizontal lines that shift slightly in late 2009, because the sampling interval for temperatures was changed from 60 seconds to 66 seconds. The eruption intervals recorded have to be multiples of 66 seconds, so occasionally you have to jump over a 60 second interval. Indeed once every ll minutes, since 66-60=6 and 66/6=11. This explains the missing entries in the minute table. (Thanks for the explanation to Dr Taylor). The same dark horizontal lines each represent many multiple points, so that it is impossible to judge changing density from them. The smaller counts for the lower intervals indeed are more revealing, and do suggest a secondary mode in the interval distribution that is increasing in frequency after 2009.

3.2 A jittered plot

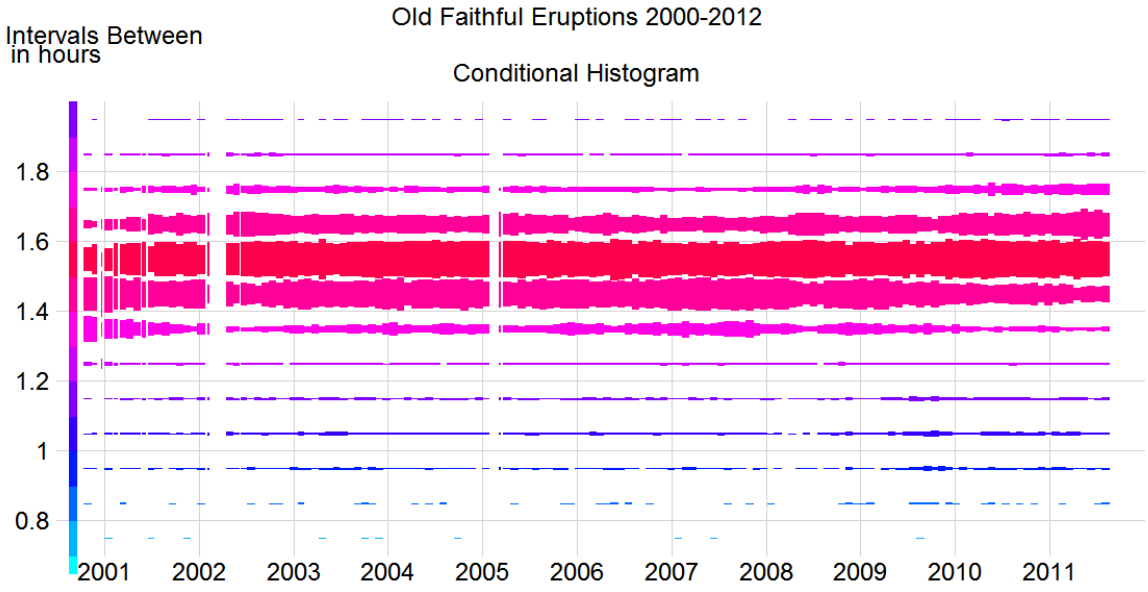
```
tiff("pictures/jitter Taylor.tif", w=1200, h=700)
lt <- sum(use)
jitter <- (runif(lt) - 0.5) * (1 / 60)
shift <- oft$time[use] > 2009.721
jitter[shift] <- jitter[shift] * 66/60

Grid(xticks=c(2000.5,2001:2012),
yticks=c(0.7,seq(0.8, 2.0, 0.2)),
ylab = "Old Faithful Eruptions 2000-2012/Intervals
Between/in hours", at=c(2006, 2001, 2000.5), cex=2.5)
points(oft$time[use], oft$interval[use]+jitter,
cex=0.2)
dev.off()
```



We jitter the plot by adding a random number uniform between -60 and 0 seconds when the sampling interval is 60 seconds, and uniform between -66 and 0 seconds when the sampling interval is 66 seconds, supposing that the actual event occurred randomly in the period between the last sampling time and the present sampling time. It is now possible to better see the overall trends, for example the increase in the location of the high density region between 2000 and 2002, and a slight increase again in 2010-2012.

3.3 Conditional Histograms of Old Faithful

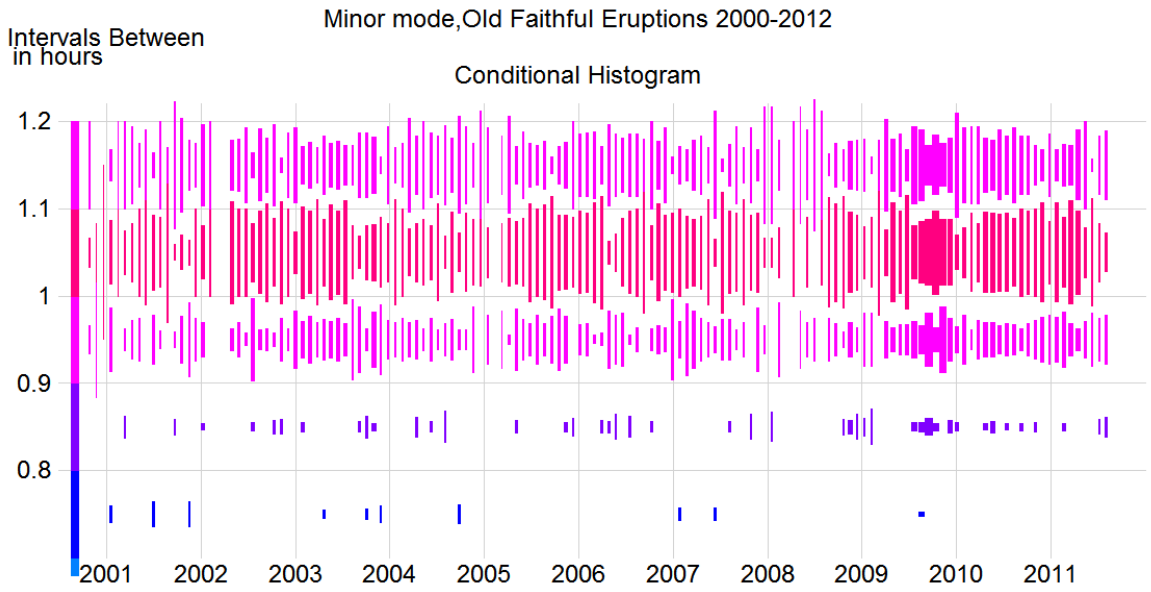
```
tiff("pictures/condHist Taylor.tif", w=1200, h=600)
Grid(xticks=c(2000.5,2001:2012),
yticks=c(0.7,seq(0.8, 2.0, 0.2)),
ylab = "Old Faithful Eruptions 2000-2012/Intervals
Between/in hours/Conditional Histogram", at=c(2006,
2001, 2000.5, 2006), cex=2)

condHist(oft$time, oft$interval, heightadj=1.3,
   xbreaks=145, ybreaks=seq(0, 2, 0.1) )
dev.off()
```



The number of eruption intervals in each 6 minute time segment is computed for each month. Aconditional histogram plots these counts as rectangles in area proportional to the count plotted at x value = month, y value = eruption interval length. A typical count would be the number of eruption intervals of length between 1 hour and 1 hour and 6 minutes in January, 2004.
The early increase in interval length between 2000 and 2002 is indicated by the declining sizes of grey blocks and the increasing sizes of black blocks. The same indicators show fluctuations throughout the period, On the whole though the fluctuations are small, and the distribution of intervals changes very little over the period.

3.4 Conditional histogram for the minor mode

```
tiff("pictures/condHistMinorModeTaylor.tif", w=1200,
h=600)
Grid(xticks=c(2000.5,2001:2012),
yticks=c(0.7,seq(0.8, 1.2, 0.1), 1.22),
ylab = "Minor mode,Old Faithful Eruptions
2000-2012/Intervals Between/in hours/Conditional
Histogram", at=c(2006, 2001, 2000.5, 2006), cex=2)

condHist(oft$time, oft$interval, heightadj=2,
   xbreaks=145, ybreaks=seq(0, 1.2, 0.1))
dev.off()
```



Looking only at the values less than 1.3 hours, we see the location distribution of the minor mode does not change significantly in the period, with values concentrated between .9 hours and 1.3 hours. In 2008-2009, the minor modes were rare; then, in mid 2009 the minor mode b. Minor mode occurrences are more frequent recently than for the whole period.

4 Old Faithful Ranger intervals 1970-2010

```
ofd <- read.csv("data/Stephens.csv", as.is=T, header=T)
head(ofd)
```

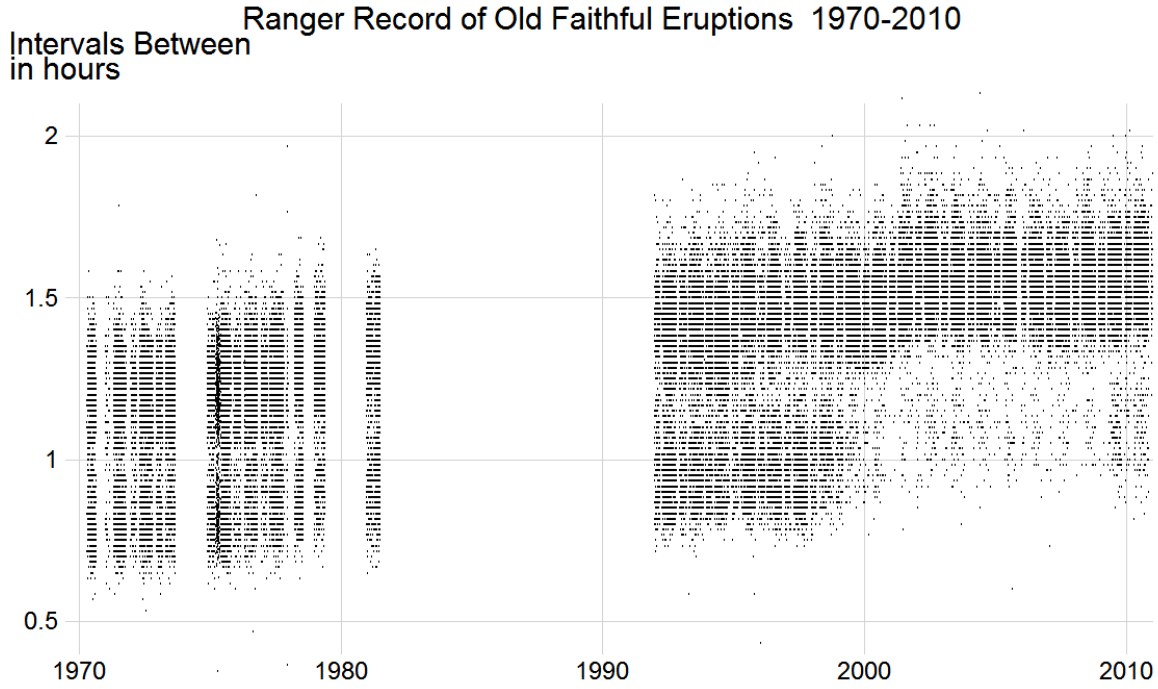
```
     Date  Start Interval Duration       years duramin
1 5/2/1970   9:40     1:11    1 1/2 1970.332610    1.50
2 5/2/1970  10:30     0:50    4 1/4 1970.332705    4.25
3 5/2/1970  11:37     1:07        4 1970.332833    4.00
4 5/2/1970 12:58p     1:21    1 3/4 1970.332987    1.75
5 5/2/1970  1:41p     0:43    4 1/2 1970.333069    4.50
6 5/2/1970  2:51p     1:10     4:00 1970.333202    4.00
      inthours   starthours
1 1.1833333333  9.666666667
2 0.8333333333 10.500000000
3 1.1166666667 11.616666667
4 1.3500000000 12.966666667
5 0.7166666667 13.683333333
6 1.1666666667 14.850000000
```

```
tail(ofd)
```

```
           Date Start Interval Duration       years
56645 12/31/2010  9:29     1:39     4:18 2011.025740
56646 12/31/2010 11:05     1:36     4:00 2011.025923
56647 12/31/2010 12:31     1:26     4:18 2011.026086
56648 12/31/2010 14:05     1:34     4:11 2011.026265
56649 12/31/2010 15:37     1:32    ~4:15 2011.026440
56650 12/31/2010 17:13     1:36     4:12 2011.026623
          duramin    inthours   starthours
56645 4.300000000 1.650000000  9.483333333
56646 4.000000000 1.600000000 11.083333333
56647 4.300000000 1.433333333 12.516666667
56648 4.183333333 1.566666667 14.083333333
56649 4.250000000 1.533333333 15.616666667
56650 4.200000000 1.600000000 17.216666667
```

4.1 Scatter plot for Ranger intervals

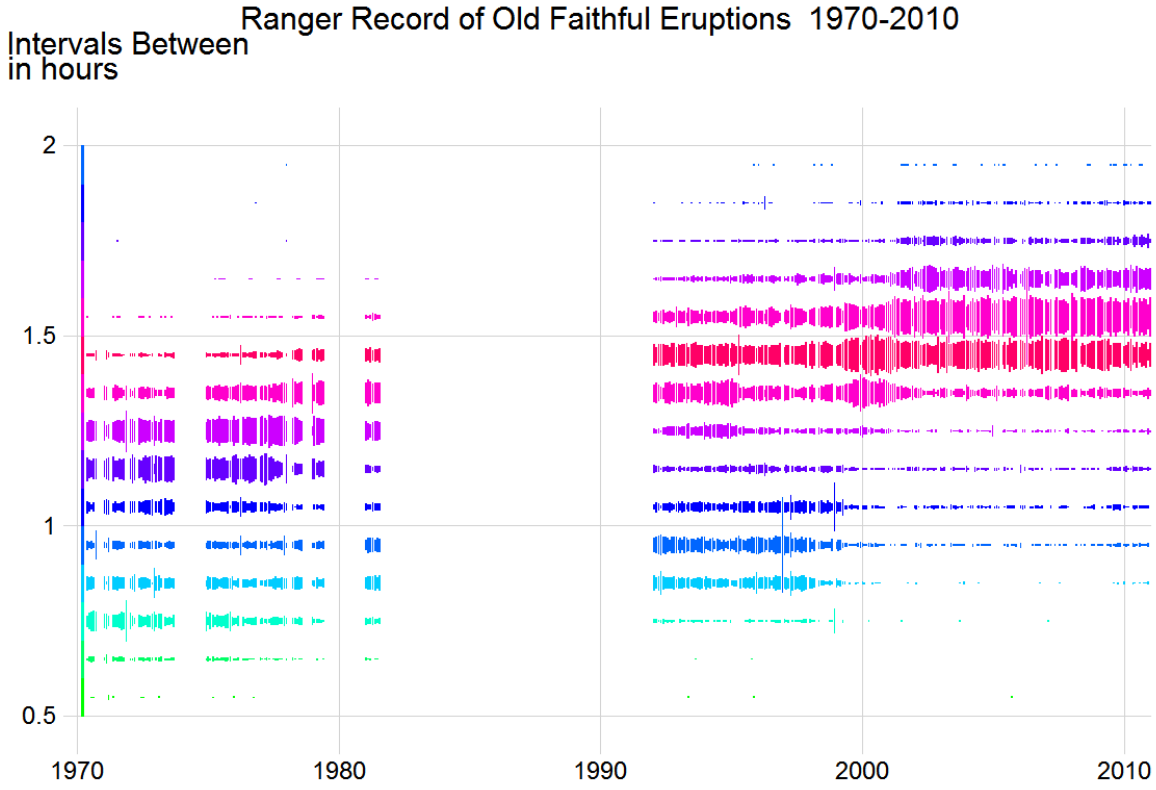
```
tiff("pictures/Ranger intervals 1970-2010.tif", w=1200,
h=700)
Grid(xticks=c(1969.5,seq(1970, 2010, 10), 2011),
yticks=c(0.4,seq(0.5, 2.0, 0.5), 2.1),
ylab = "Ranger Record of Old Faithful Eruptions
1970-2010/Intervals Between/in hours", at=c(1990, 1972,
1969.5), cex=2.5)

points(ofd$years, ofd$inthours, cex=0.2)
dev.off()
```



The vertical white columns correspond to missing years; the thin white vertical stripes correspond to missing weeks in November and March when the park personnel are on furlough. There are two clear modes in the interval distributions. The minor mode almost disappeared after 1998, perhaps due to a local earthquake in 1998. (There is a stream of small and moderate earthquakes in this giant hollow squishy region, so who knows.) The minor mode starts to reappear in 2009 and 2010.

4.2 Conditional Histogram Ranger Intervals

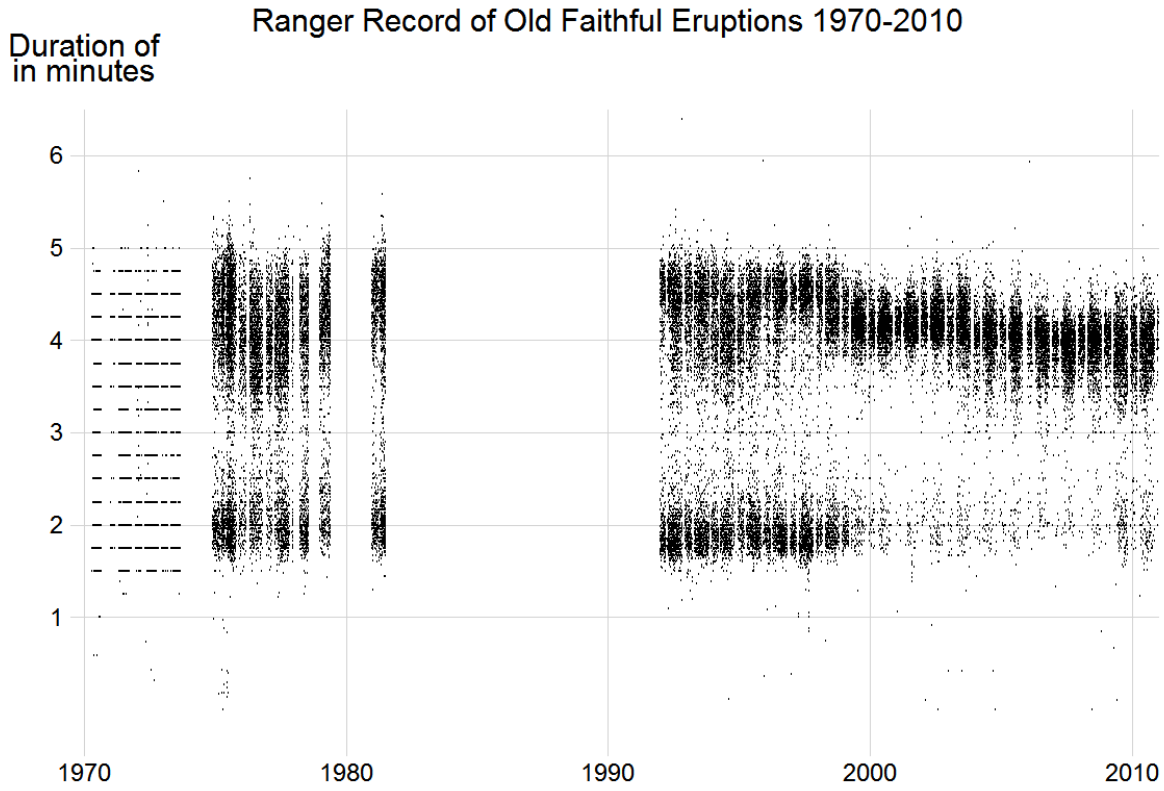
```
tiff("pictures/CondHist for intervals 1970-2010.tif",
w=1200, h=800)
Grid(xticks=c(1969.5,seq(1970, 2010, 10), 2011),
 yticks=c(0.4,seq(0.5, 2.0, 0.5), 2.1),
ylab = "Ranger Record of Old Faithful Eruptions
1970-2010/Intervals Between/in hours", at=c(1990, 1972,
1969.5), cex=2.5)

condHist(ofd$years,  ofd$inthours, xlab="years",
ylab="intervals", xbreaks=493, ybreaks=seq(0.5, 2,
0.1), heightadj=2.5)
dev.off()
```



The monthly histograms show the increase in interval length between 1970-1990 and 2000-2010. The major mode has a maximum at 1hour 15 minutes in the first period, and at 1 hour 35 minutes in the second period. The minor mode interval increased between 1970 and 2000. The minor mode almost disappeared between 2000 and 2010.

5 Ranger durations
5.1 Scatter plot Ranger Durations

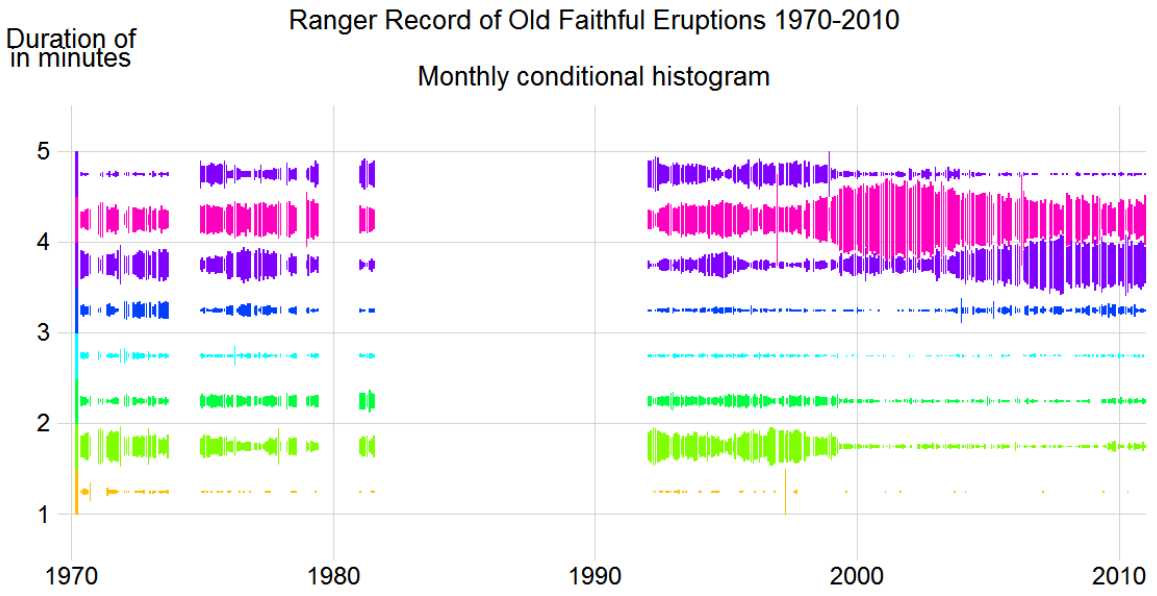
```
tiff("pictures/Ranger durations 1970-2010.tif", w=1200,
h=800)
Grid(xticks=c(1969.5,seq(1970, 2010, 10), 2011),
 yticks=c(-0.5,1:6, 6.5),
ylab = "Ranger Record of Old Faithful Eruptions
1970-2010/Duration of/in minutes", at=c(1990, 1970,
1970), cex=2.5)

points(ofd$years, ofd$duramin, cex=0.2)
dev.off()
```



The straight horizontal lines in the 70's are due to the use of fractional minutes (1/2,1/4,3/4) rather than minutes and seconds. There are two very distinct modes, although the minor mode almost disappears in 1998 as for the intervals. It appears more frequently in 2009 and 2010. The major mode duration has declined over the period.

5.2 Conditional histogram of Ranger durations

```
tiff("pictures/Monthly histogram of Ranger
Durations.tif", w=1200, h=600)
Grid(xticks=c(1969.5,seq(1970, 2010, 10), 2011),
yticks=c(0.5,1:5, 5.5),
ylab = "Ranger Record of Old Faithful Eruptions
1970-2010/Duration of/in minutes/Monthly conditional
histogram", at=c(1990, 1970, 1970, 1990), cex=2.2)
condHist(ofd$years,  ofd$duramin,
     xbreaks=493, ybreaks=seq(1, 5, 0.5), heightadj=2)
dev.off()
```



The major mode peaks at about 4 minutes, and the minor mode peaks at about 2 minutes throughout the whole period. The durations are quite hard to measure, since a subjective choice must be made about when the eruption starts (there is a long period of foreplay when smallish amounts of water and steam appear intermittently, and a similar period of postplay). Both 4 minutes and 2 minutes appear far more frequently than the durations a few seconds away from those numbers.

It seems as if there was very little change in location or frequency of either mode between 1970 and 1998. But then the minor mode nearly disappears, and the major mode duration declines a little, and becomes more concentrated.

6 Old Faithful Intervals vs duration, 1970-2010

Eliminate outliers in duration and interval:

```
ofdnew <- ofd[ofd$duramin < 5 & ofd$inthours < 2, ]
```

Identify previous durations for predicting intervals:

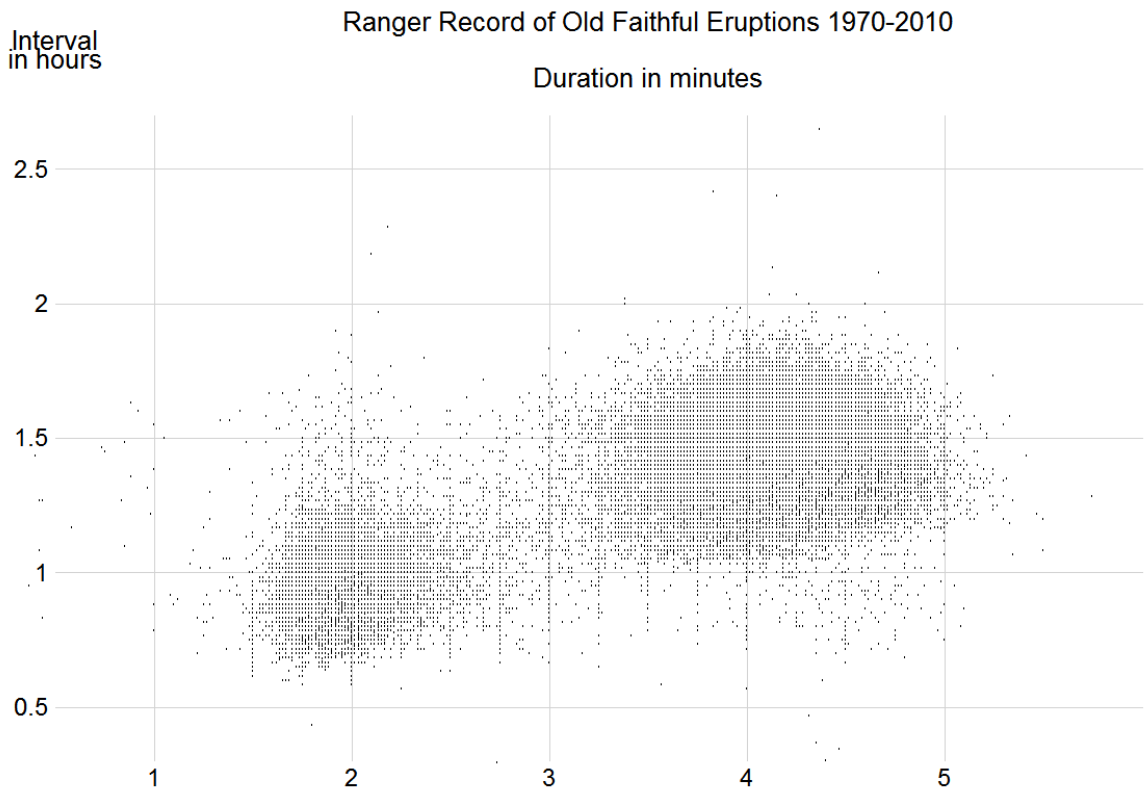
```
duration <- ofd$duramin[-dim(ofd)[1]]
interval <- ofd$inthours[-1]
interval[diff(ofd$years) > 3/(24 * 365)] <- NA
```

Identify period after 1999 when changes in mode locations and frequencies have occurred, as being more relevant for ranger predictions:

```
when <- (ofd$years>1999)[-1]
```

6.1 Scatter plot, Intervals vs Duration 1970-2010

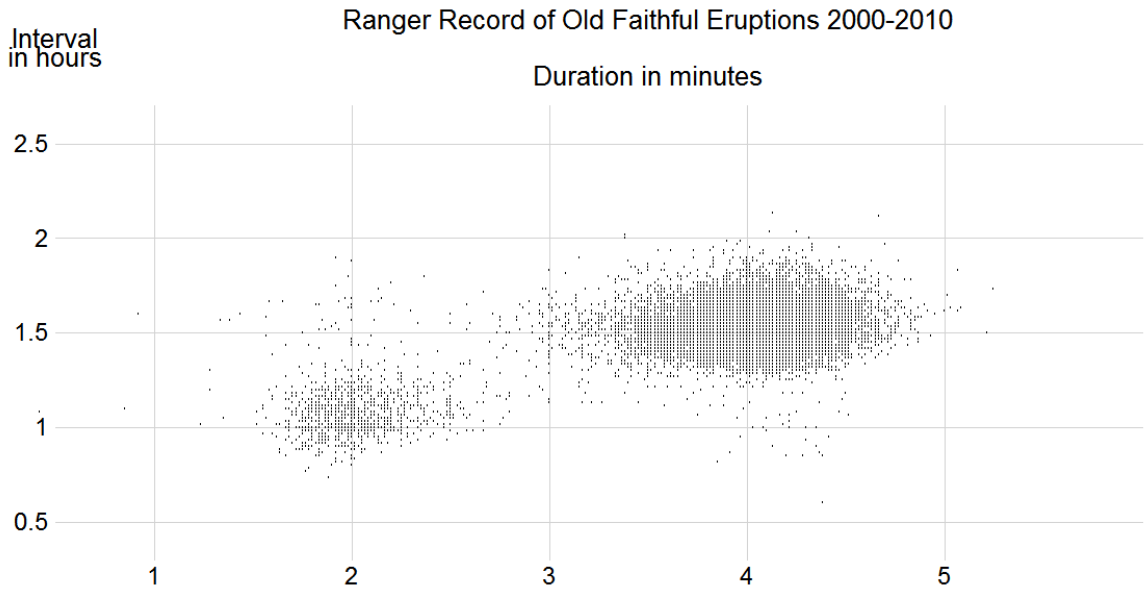
```
tiff("pictures/Intervals vs durations.tif", w=1200,
h=800)
Grid(xticks=c(0.5,1:6),
 yticks=c(0.3,seq(0.5,2.5,0.5), 2.7),
ylab = "Ranger Record of Old Faithful Eruptions
1970-2010/Interval/in hours/Duration in minutes",
at=c(3.5, 0.5, 0.5, 3.5), cex=2.2)

points(duration, interval, cex=0.2)
dev.off()
```



The distinct horizontal lines come from the relatively coarse measurement of duration in the 70's. The grid appearance is caused by the relatively coarse measurements in both variables. There are many repeated values at each point, so we do not have a good picture of density variation. Still, there are two very distinct modes, supporting the original idea of regression to predict interval from duration, or the newer idea of low duration predicts low interval.

6.2 Old Faithful Intervals vs duration, 2000-2010

```
tiff("pictures/Intervals vs durations 2000-2010",
w=1200, h=600)
Grid(xticks=c(0.5,1:6),
 yticks=c(0.3, seq(0.5, 2.5, 0.5), 2.7),
ylab = "Ranger Record of Old Faithful Eruptions
2000-2010/Interval/in hours/Duration in minutes",
at=c(3.5, 0.5, 0.5, 3.5), cex=2.2)

points(duration[when], interval[when], cex=0.2)
dev.off()
```



We see two flat ellipses corresponding to observations near either the minor mode or the major mode, the minor mode for durations less than 3 minutes, the major mode for durations greater than 3 minutes.

```
sum(duration[when] < 3)
```

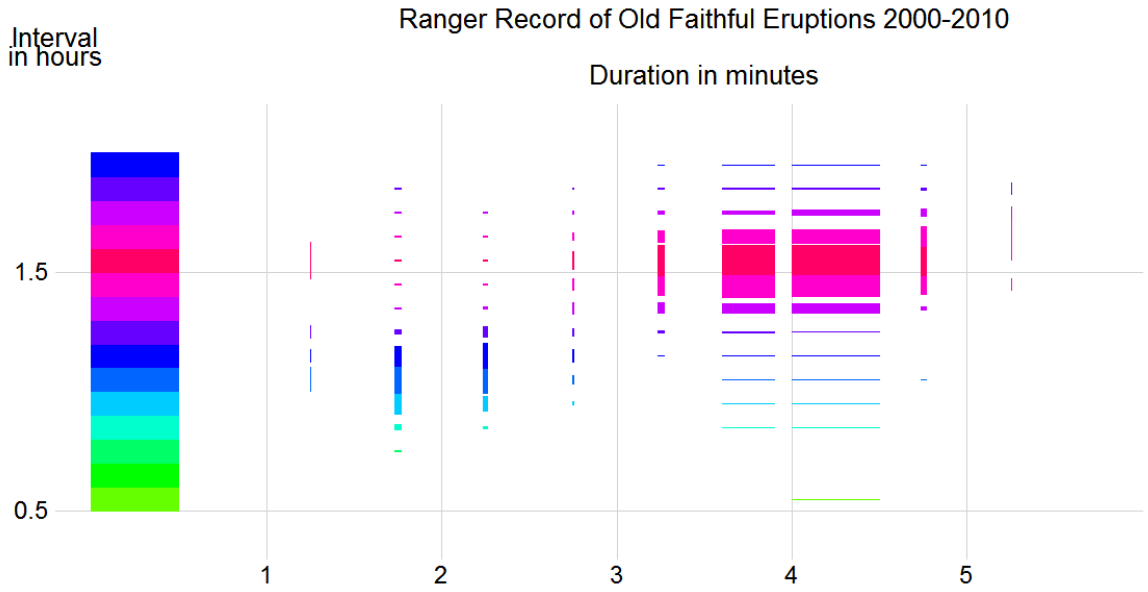
```
[1] 1238
```

```
sum(duration[when] > 3)
```

```
[1] 19438
```

6.3 Conditional hist for recent Interval on Duration

```
tiff("pictures/interval~duration.tif",w=1200, h=600)
Grid(xticks=c(-0.2,1:6), yticks=c(0.3,seq(0.5,2),
2.2),
ylab = "Ranger Record of Old Faithful Eruptions
2000-2010/Interval/in hours/Duration in minutes",
at=c(3.5, -0.2, -0.2, 3.5), cex=2.2)

condHist(duration[when],  interval[when], heightadj=2,
     xbreaks=seq(1, 6, 0.5), ybreaks=seq(0.5, 2, 0.1))
dev.off()
```



We see constant interval distributions within the very infrequent minor mode (duration<3) and within the major mode(duration>3)

7 What is a ranger to do?

We compare using a step function and a regression line for the prediction of interval length from duration.
We compare the estimated standard deviation of the error in using the step function, compared to the error in using regression, in the period 1999-2010.

First, for the step function, with the step at 3 mins duration:

```
dw <- duration[when]
iw <- interval[when]
use <- !is.na(dw) & !is.na(iw)
iw <- iw[use]
dw <- dw[use]
cut <- dw < 3
round(sqrt((sd(iw[cut])^2*sum(cut)+sd(iw[!cut])^2*
sum(!cut)) / length(iw) ), 3)
```

```
[1] 0.119
```

Next, for the regression

```
round(summary(lm(iw~dw))$sigma, 3)
```

```
[1] 0.125
```

From which we conclude that the step function gives a slightly more accurate prediction than the straight line, with a standard error of .119 * 60 = 7 minutes

8 Suggested Ranger Action

If the guy has a beard, don't talk to him. He will suggest you make predictions in some entirely different way.
Otherwise: The conditional distribution of the next interval doesn't change for durations >3 mins, and it is also nearly unchanged for durations <3 mins, and so you don't need regression.
The prediction is:
duration < 3 mins: predict next interval 1 hour and 10 mins,
duration >= 3 mins: predict next interval 1 hour and 30 minutes.

The minor mode, duration < 3 mins, only occurred 6% of the time in 2000-2010, but it is appearing more frequently recently.

9 Data Preparation

9.1 Scraping the Taylor data

Scraping is a general term for converting web data from its presented form to a form you can do the analysis in. In statistical analysis, the standard form is a data frame with named columns corresponding to variables, and rows corresponding to individual instances for which those variables are measured. The Taylor data resides in 12 data sets corresponding to the years 2000-2011. For the year 2000 the file is: { http://www.geyserstudy.org/geysers/OLDFAITHFUL/eruptions/Old%20Faithful%20eruptions%20for%202000.TXT} .

9.11 Scan in the files:

To access the data we use the general purpose R reader scan. Scan reads each file one line at a time into a vector of strings, one string per line. The vectors are combined for all 12 files by looping the read over the files in order, pasting the year separately into the standard file name given above.

```
if(FALSE){
oftr=""
base <-
"http://www.geyserstudy.org/geysers/OLDFAITHFUL/eruptions/Old%20Faithful%20eruptions%20for%20"
for ( year in 2000:2011){
file <- paste(base,  year, ".TXT", sep="")
thisof <- scan(file, what="", sep="\n", quiet=T)
oftr=c(oftr, thisof)
}

# Save the raw data to avoid accessing the web pages again:
cat(oftr, file="data/Raw Taylor", sep="\n")
}
```

Start later runs here:

```
oftr <- scan("data/Raw Taylor", what="", sep="\n",
quiet=T)
head(oftr)
```

```
[1] "Old Faithful Geyser Eruptions"
[2] "  Time and Date      Interval"
[3] "  "
[4] "10/11/00 13:56:55,  0:00:00"
[5] "10/11/00 15:17:55,  1:21:00"
[6] "10/11/00 16:40:55,  1:23:00"
```

The first six lines of the file show some header material, then lines consisting of dates, times of eruption, and intervals between eruptions. These lines containing numbers are the lines we are interested in, so we select these using the character pattern searcher grep. Some further lines are eliminated containing the word "eruption".

```
oftn <- oftr[grep("[0-9]", oftr)]
oftn <- oftn[!grepl("eruption", oftn)]
oftn[1:10]
```

```
[1] "10/11/00 13:56:55,  0:00:00"
 [2] "10/11/00 15:17:55,  1:21:00"
 [3] "10/11/00 16:40:55,  1:23:00"
 [4] "10/11/00 18:09:55,  1:29:00"
 [5] "10/11/00 19:30:55,  1:21:00"
 [6] "10/11/00 21:00:55,  1:30:00"
 [7] "10/11/00 22:21:55,  1:21:00"
 [8] "10/11/00 23:44:55,  1:23:00"
 [9] "10/12/00 01:19:55,  1:35:00"
[10] "10/12/00 02:55:55,  1:36:00"
```

9.12 Convert time information to fractional years:

Unlike in many web based presentations, the measured values are all in the same position on each line of the file, so we can select them simply. We get the year and eruption time information first, to set up a time for each interval. The numbers have to be converted from characters to numbers, and suitably combined to get a single number measuring eruption time down to the minute.

```
years <- as.numeric(substr(oftn, 7, 8))
months <- as.numeric(substr(oftn, 1, 2))
days <- as.numeric(substr(oftn, 4, 5))
```

```
hours <- as.numeric(substr(oftn, 10, 11))
minutes <- as.numeric(substr(oftn, 13, 14))
daymonth <- cumsum(c(0, 31, 29, 31, 30, 31, 30, 31, 31,
30, 31, 31))
time <- 2000 + years + daymonth[months]/366  +
   (days - 1)/366 + hours /(24 * 366) +
    minutes /(24 * 366 * 60)
```

Finally select the hours and minutes for the interval variable, and convert into a single numerical variable, eliminating some bad values:

```
interval <- as.numeric(substr(oftn, 21, 21)) +
        as.numeric(substr(oftn, 23, 24)) / 60 +
        as.numeric(substr(oftn, 26, 27)) / 3600
oft <- data.frame(time=time, interval=interval)
oft <- oft[interval!=0 & interval <4 & !is.na(interval),]
```

Usually there is a fair bit of nasty detailed particular unpredictable idiosyncratic work identifying bad values, changing formats, and whole bad passages, correcting them from the original data, or declaring them missing if it is not possible to determine the correct value. Here, only 33 of 58516 values required correction.
headsortunique is a convenient function to see what is there:

```
hsu <- function(x, n=20) head(sort(unique(x)), n)
```

For example we examine the interval minutes in the years 2010 and 2011 by:

```
hsu(substr(oftn[years > 9], 23, 24), 60)
```

```
[1] "00" "01" "02" "03" "04" "05" "06" "07" "08" "09" "10"
[12] "11" "12" "13" "14" "15" "17" "18" "19" "20" "21" "22"
[23] "23" "24" "25" "26" "28" "29" "30" "31" "32" "33" "34"
[34] "35" "36" "37" "39" "40" "41" "42" "43" "44" "45" "46"
[45] "47" "48" "50" "51" "52" "53" "54" "55" "56" "57" "58"
[56] "59"
```

This is interesting because there are no occurrences of minutes 16, 27, 38, or 49 and only one 5. All differing by 11! A solution will appear.

9.13 Save what you have done:

```
write.csv(oft, "data/Taylor.csv", row.names=F)
```

9.2 Scraping and Shaping the Stephens data

From { http://www.geyserstudy.org/ofvclogs.aspx}
These data are transcribed by Lynn Stephens, Marion Powell, and Mary Schwarz, from the ranger logs of Old Faithful eruption intervals and durations, from 1970 to 2010. They are usually only available during the working day.

The 1970 data appears in the file: { http://www.geyserstudy.org/ofvclogs/log1970.txt}

9.21 Loop through years:

We collect all the data by looping over the years for which data have been transcribed:

```
if(FALSE){
of <- ""
for ( year in c( 1970:1973, 1975:1979, 1981, 1992:2010)
){
file <-
paste("http://www.geyserstudy.org/ofvclogs/log",
              year, ".txt", sep="")
# scan data into vector of strings
thisof <- scan(file, what="", sep="\n", quiet=T)
of <- c(of, thisof)
}
# Select only Old Faithful entries:
gof <- grep("Old", of, value=TRUE)
head(gof)

#Save the raw data to avoid having to repeatedly access
web files:
cat(gof, file="data/RawRanger", sep="\n")
}
```

9.22 Restart calculations here: entries are tab separated:

```
gof <- read.delim("data/Raw Ranger", header=F, as.is=T,
sep="\t")
head(gof, 5)
```

```
       V1           V2     V3 V4   V5    V6 V7 V8 V9 V10
1 5/2/1970 Old Faithful   8:29          4:00
2 5/2/1970 Old Faithful   9:40    1:11 1 1/2
3 5/2/1970 Old Faithful  10:30    0:50 4 1/4
4 5/2/1970 Old Faithful  11:37    1:07     4
5 5/2/1970 Old Faithful 12:58p    1:21 1 3/4
  V11
1
2
3
4
5
```

```
tail(gof, 5)
```

```
             V1           V2    V3 V4   V5    V6    V7
87206 12/31/2010 Old Faithful 12:31    1:26  4:18 12:28
87207 12/31/2010 Old Faithful 14:05    1:34  4:11 13:59
87208 12/31/2010 Old Faithful 15:37    1:32 ~4:15 15:36
87209 12/31/2010 Old Faithful 17:13    1:36  4:12 17:00
87210 12/31/2010 Old Faithful 23:25 vr
        V8    V9 V10 V11
87206 130+ 12:38
87207      14:04
87208  145 15:38
87209      17:10
87210
```

The entries are dates, times of eruption, interval between times, and duration. The durations are sometimes given in minutes and seconds, and other times in minutes and fractions. There is considerable variation in the format in which the various numbers are expressed. There are many extraneous characters and verbal comments from the rangers.

The entries 1, 3, 5, 6 contain respectively the date, the time of eruption, the interval between eruptions, and the duration. We pick off these entries, put them in a matrix gom, and then check the contents of gom:

```
gom <- gof[, c(1, 3, 5, 6)]
names(gom) <- c("Date", "Start", "Interval", "Duration")
head(gom)
```

```
     Date  Start Interval Duration
1 5/2/1970   8:29             4:00
2 5/2/1970   9:40     1:11    1 1/2
3 5/2/1970  10:30     0:50    4 1/4
4 5/2/1970  11:37     1:07        4
5 5/2/1970 12:58p     1:21    1 3/4
6 5/2/1970  1:41p     0:43    4 1/2
```

```
tail(gom)
```

```
           Date Start Interval Duration
87205 12/31/2010 11:05     1:36     4:00
87206 12/31/2010 12:31     1:26     4:18
87207 12/31/2010 14:05     1:34     4:11
87208 12/31/2010 15:37     1:32    ~4:15
87209 12/31/2010 17:13     1:36     4:12
87210 12/31/2010 23:25
```

```
dim(gom)
```

```
[1] 87210     4
```

9.23 Convert date to years:

Correct some bad dates out of order in days plot:

```
gom[gom[, 1]=="8/5/2003" & gom[, 2] == "15:54", 1] <-
"8/5/1977"
gom[, 1] <- sub("/92", "/1992", gom[, 1])
gom[, 1] <- sub("/76", "/1976", gom[, 1])
gom[, 1] <- sub("/2020", "/2010", gom[, 1])
gom[, 1] <- sub("/1982", "/1992", gom[, 1])
```

Convert character dates to numerical years:

```
gom$years <- 1970+julian( as.Date(gom[, 1],
"%m/%d/%Y"))/365
gom[8:12, ]
```

```
      Date Start Interval Duration       years
8  5/2/1970 3:57p     1:06     4:00 1970.331507
9  5/2/1970 5:06p     1:09        ? 1970.331507
10 5/3/1970  8:38                4:00 1970.334247
11 5/3/1970  9:48     1:10     2:00 1970.334247
12 5/3/1970 10:32     0:44    4 1/4 1970.334247
```

9.24 Fix Durations, convert to seconds:

Fix all the horrible possibilities seen in head sort unique:

```
x <- gom$Duration
hsu(x, 20)
```

```
[1] ""                "---"              " "
 [4] "          ?"     "          1 (+)" "          2?"
 [7] "          3??"   "          4?"    "         ?"
[10] "         2?"     "         3???"   "         4+"
[13] "        4+"      "       ~4"       "       3 1/2?"
[16] "    long"        "   long?"        "  long"
[19] " ?"              " ~1 3/4"
```

```
x <- gsub("1/4", ":15", x)
x <- gsub("3/4", ":45", x)
x <- gsub("1/2", ":30", x)
x <- gsub("1/3", ":20", x)
```

Accept only digits, :, and blanks:

```
x <- gsub("[^0-9:]", "", x)
hsu(x, 20)
```

```
[1] ""          ":"         ":00"       ":06"
 [5] ":21700:23" ":35"        ":4:06"     ":51"
 [9] ":7"        "0:00"      "0:22"      "0:3:11"
[13] "0:3:25"    "0:3:50"    "0:3:58"    "0:3:59"
[17] "0:4:00"    "0:4:08"    "0:4:10"    "0:4:14"
```

Get two colons down to 1 if 1 digit follows first colon

```
u <- grepl(".*:[0-9]:", x)
x[u] <- sub(".*:", "", x[u])
hsu(x, 20)
```

```
[1] ""          ":"         ":00"       ":06"
 [5] ":21700:23" ":35"        ":51"       ":7"
 [9] "0:00"      "0:22"       "0:51"      "0:53"
[13] "00"        "00:00"      "00:26"     "00:36"
[17] "01:01"     "01:02"      "01:05"     "01:13"
```

Reduce other two colon cases to single colon:

```
u <- grepl(":.*:", x)
x[u] <- " : "
hsu(x, 20)
```

```
[1] ""      " : "   ":"     ":00"   ":06"   ":35"   ":51"
 [8] ":7"    "0:00"  "0:22"  "0:51"  "0:53"  "00"    "00:00"
[15] "00:26" "00:36" "01:01" "01:02" "01:05" "01:13"
```

Fix all single colons to have spaces on each side:

```
u <- nchar(x)==1 & grepl(":", x)
x[u] <- " : "
hsu(x, 20)
```

```
[1] ""      " : "   ":00"   ":06"   ":35"   ":51"   ":7"
 [8] "0:00"  "0:22"  "0:51"  "0:53"  "00"    "00:00" "00:26"
[15] "00:36" "01:01" "01:02" "01:05" "01:13" "01:16"
```

Get double digits into 00:seconds format

```
u <- (nchar(x) == 2) & grepl("[0-9][0-9]", x)
x[u] <- paste("00:", x[u], sep="")
hsu(x, 20)
```

```
[1] ""      " : "   ":00"   ":06"   ":35"   ":51"   ":7"
 [8] "0:00"  "0:22"  "0:51"  "0:53"  "00:00" "00:06" "00:08"
[15] "00:10" "00:11" "00:14" "00:15" "00:16" "00:18"
```

Fix starting ":" :

```
u <-substr(x, 1, 1) == ":" & nchar(x) > 1
x[u] <- paste("00", x[u], sep="")
hsu(x, 20)
```

```
[1] ""      " : "   "0:00"  "0:22"  "0:51"  "0:53"  "00:00"
 [8] "00:06" "00:08" "00:10" "00:11" "00:14" "00:15" "00:16"
[15] "00:18" "00:19" "00:20" "00:23" "00:24" "00:25"
```

Get single digits into :00 format

```
u <- (nchar(x) == 1) & grepl("[0-9]", x)
x[u] <- paste(x[u], ":00", sep="")
hsu(x, 20)
```

```
[1] ""      " : "   "0:00"  "0:22"  "0:51"  "0:53"  "00:00"
 [8] "00:06" "00:08" "00:10" "00:11" "00:14" "00:15" "00:16"
[15] "00:18" "00:19" "00:20" "00:23" "00:24" "00:25"
```

True, we just lost 26000 observations in which the durations were missing! More could be extracted with a more pains-taking search.

Fix ending and beginning colons:

```
u <-substr(x, nchar(x), nchar(x)) == ":"
x[u] <- paste( x[u], "00", sep="")
u <- substr(x, 1, 1) ==":"
x[u] <- paste( "00",  x[u],  sep="")
```

Pick off minutes and seconds, first getting single ":" :

```
u <- grepl(":", x)
sum(u)
```

```
[1] 61151
```

```
mt <- rep(NA, length(x))
st <- mt
xl <- unlist( strsplit(x[u], ":") )
length(xl)
```

```
[1] 122302
```

```
head(xl)
```

```
[1] "4"  "00" "1"  "30" "4"  "15"
```

```
mt[u] <- as.numeric( xl[ seq(1, length(xl), 2)] )
st[u] <- as.numeric( xl[ seq(2, length(xl), 2)] )
```

Remove dubious seconds:

```
hsu(mt[u], 60)
```

```
[1]     0     1     2     3     4     5     6     7     8
[10]    9    10    11    12    13    14    15    16    17
[19]   18    19    20    21    22    23    29    43    44
[28]  214   803  1600  2009  2317  3014  7518 10012 14313
```

```
hsu(st[u], 60)
```

```
[1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[19] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
[37] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
[55] 54 55 56 57 58 59
```

```
notna <- !is.na(st[u])
st[u][notna][st[u][notna] > 59] <- NA
mt[u] <- st[u]/60 +  mt[u]
```

Remove the few durations > 10 mins, after looking at plot:

```
notna <- !is.na(mt)
mt[notna][ mt[notna] > 10 | mt[notna] < 0] <- NA
gom$duramin <- mt
head(gom)
```

```
     Date  Start Interval Duration       years duramin
1 5/2/1970   8:29              4:00 1970.331507    4.00
2 5/2/1970   9:40     1:11    1 1/2 1970.331507    1.50
3 5/2/1970  10:30     0:50    4 1/4 1970.331507    4.25
4 5/2/1970  11:37     1:07        4 1970.331507    4.00
5 5/2/1970 12:58p     1:21    1 3/4 1970.331507    1.75
6 5/2/1970  1:41p     0:43    4 1/2 1970.331507    4.50
```

```
dim(na.omit(gom))
```

```
[1] 60945     6
```

The rule for this kind of data is that if any bad thing could happen, it did.

9.25 Fix interval, convert to hours:

These fixes are similar to durations. The intervals need to be gotten into format d:dd:dd

```
x <- gom$Interval
```

Accept only digits, or ":":

```
x <- gsub("[^0-9:]", "", x)
hsu(x, 20)
```

```
[1] ""        ":51"     ":610"    "0"       "0:00"
 [6] "0:03"    "0:11:45" "0:17:35" "0:18:09" "0:20:45"
[11] "0:22"    "0:26"    "0:28"    "0:34"    "0:35"
[16] "0:36"    "0:36:00" "0:37"    "0:37:00" "0:38"
```

Fix 0:d:dd cases

```
u <- grepl("0:[0-9]:[0-9][0-9]", x)
x[u] <- substr(x[u], 3, 6)
hsu(x, 20)
```

```
[1] ""        ":51"     ":610"    "0"       "0:00"
 [6] "0:03"    "0:11:45" "0:17:35" "0:18:09" "0:20:45"
[11] "0:22"    "0:26"    "0:28"    "0:34"    "0:35"
[16] "0:36"    "0:36:00" "0:37"    "0:37:00" "0:38"
```

Fix double digit minutes without ":"(data given in minutes):

```
u <- grepl("[0-9][0-9]",x) & ! grepl(":" ,x)
x[u] <- paste("00", x[u], "00", sep=":")
hsu(x, 20)
```

```
[1] ""        ":51"     ":610"    "0"       "0:00"
 [6] "0:03"    "0:11:45" "0:17:35" "0:18:09" "0:20:45"
[11] "0:22"    "0:26"    "0:28"    "0:34"    "0:35"
[16] "0:36"    "0:36:00" "0:37"    "0:37:00" "0:38"
```

Fix single digits:

```
u <- grepl("[0-9]", x) &! grepl(":", x)
x[u] <- paste(x[u],"00", "00", sep=":")
hsu(x, 20)
```

```
[1] ""        ":51"     ":610"    "0:00"    "0:00:00"
 [6] "0:03"    "0:11:45" "0:17:35" "0:18:09" "0:20:45"
[11] "0:22"    "0:26"    "0:28"    "0:34"    "0:35"
[16] "0:36"    "0:36:00" "0:37"    "0:37:00" "0:38"
```

Fix initial ":":

```
u <- substr(x, 1, 1)==":"
x[u] <- paste("00", x[u], "")
```

Fix single : values to double :

```
u <- grepl(":", x) & ! grepl(":.*:", x)
x[u] <- paste(x[u], ":00", sep="")
```

Fix any remaining values to double :

```
u <- ! grepl(":", x)
x[u] <- paste(x[u], " : : ", sep="")
hsu(x, 20)
```

```
[1] " : : "   "0:00:00" "0:03:00" "0:11:45" "0:17:35"
 [6] "0:18:09" "0:20:45" "0:22:00" "0:26:00" "0:28:00"
[11] "0:34:00" "0:35:00" "0:36:00" "0:37:00" "0:38:00"
[16] "0:38:13" "0:39:00" "0:39:50" "0:40:00" "0:40:10"
```

Convert to hours:

```
xl <- unlist(strsplit(x, ":"))
hsu(x, 20)
```

```
[1] " : : "   "0:00:00" "0:03:00" "0:11:45" "0:17:35"
 [6] "0:18:09" "0:20:45" "0:22:00" "0:26:00" "0:28:00"
[11] "0:34:00" "0:35:00" "0:36:00" "0:37:00" "0:38:00"
[16] "0:38:13" "0:39:00" "0:39:50" "0:40:00" "0:40:10"
```

```
hours <- as.numeric( xl[seq(1, length(xl), 3)] )
mins  <- as.numeric( xl[seq(2, length(xl), 3)] )
secs  <- as.numeric( xl[seq(3, length(xl), 3)] )
hsu(hours)
```

```
[1]   0   1   2   3   4   5   6   7   8   9  10  12  13  14
[15]  15  16  27 111 311 741
```

```
hsu(mins, 60)
```

```
[1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[19] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
[37] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
[55] 54 55 56 57 58 59
```

```
summary(mins)
```

```
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
 0.00000  20.00000  29.00000  30.89606  40.00000 809.00000
    NA's
   13517
```

```
hsu(secs, 60)

 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[19] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
[37] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
[55] 54 55 56 57 58 59
```

Remove dubious mins and hours:

```
notna <- !is.na(mins)
mins[notna][mins[notna] > 95] <- NA
notna <- !is.na(hours)
```

Remove the few intervals > 4 hours, some of them genuine:

```
hours[notna][hours[notna] > 4] <- NA
gom$inthours <- hours + mins/60 + secs/3600
 head(gom)
```

```
     Date  Start Interval Duration       years duramin
1 5/2/1970   8:29              4:00 1970.331507    4.00
2 5/2/1970   9:40     1:11    1 1/2 1970.331507    1.50
3 5/2/1970  10:30     0:50    4 1/4 1970.331507    4.25
4 5/2/1970  11:37     1:07        4 1970.331507    4.00
5 5/2/1970 12:58p     1:21    1 3/4 1970.331507    1.75
6 5/2/1970  1:41p     0:43    4 1/2 1970.331507    4.50
      inthours
1           NA
2 1.1833333333
3 0.8333333333
4 1.1166666667
5 1.3500000000
6 0.7166666667
```

```
dim(na.omit(gom))
```

```
[1] 56653     7
```

9.26 Fix Start times, convert to hours:

Fix start times, as with interval:

```
x <- gom$Start
hsu(x, 10)
```

```
[1] ""
 [2] " "
 [3] "      ~07:25"
 [4] "    07:02??"
 [5] "(1) [Note:  appears that predictions for 6:59, 7:54, 9:14
eruptions were"
 [6] "(2) [on the wrong lines.]"
 [7] "**"
 [8] "*12:06 used 85 minutes"
 [9] "*20:11 preplay = 38 seconds"
[10] "*There was a pause of water and about 30 seconds later 2 more
spurts"
```

Find p character to indicate afternoon, to use later:

```
pm <- grepl("[0-9][p|P]", x)
sum(pm)
```

```
[1] 2143
```

```
x <- gsub("p|P", "", x)
hsu(x, 10)
```

```
[1] ""
 [2] " "
 [3] "      ~07:25"
 [4] "    07:02??"
 [5] "(1) [Note:  aears that redictions for 6:59, 7:54, 9:14
erutions were"
 [6] "(2) [on the wrong lines.]"
 [7] "**"
 [8] "*12:06 used 85 minutes"
 [9] "*20:11 relay = 38 seconds"
[10] "*There was a ause of water and about 30 seconds later 2 more
surts"
```

Accept only digits, or ":":

```
x <- gsub("[^0-9:]", "", x)
hsu(x, 10)
```

```
[1] ""        ":"       "0:00"    "0:00:00" "0:00:21"
 [6] "0:01"    "0:01:00" "0:02"    "0:03"    "0:03:00"
```

Remove triple :

```
u <- grepl(":.*:.*:", x)
x[u] <-" :: "
hsu(x, 10)
```

```
[1] ""        " :: "    ":"       "0:00"    "0:00:00"
 [6] "0:00:21" "0:01"    "0:01:00" "0:02"    "0:03"
```

Fix initial ":":

```
u <- substr(x, 1, 1)==":"
x[u] <- paste("00", x[u], "")
```

Fix single : values to double :

```
u <- grepl(":", x) & ! grepl(":.*:", x)
x[u] <- paste(x[u], ":00", sep="")
hsu(x, 10)
```

```
[1] ""        " :: "    "0:00:00" "0:00:21" "0:01:00"
 [6] "0:02:00" "0:03:00" "0:03:21" "0:04:00" "0:05:00"
```

Fix any remaining values to double :

```
u <- ! grepl(":", x)
x[u] <- paste(x[u], " : : ", sep="")
hsu(x, 10)
```

```
[1] " : : "   " :: "    "0:00:00" "0:00:21" "0:01:00"
 [6] "0:02:00" "0:03:00" "0:03:21" "0:04:00" "0:05:00"
```

Convert to hours

```
xl <- unlist(strsplit(x, ":"))
hours <- as.numeric( xl[seq(1, length(xl), 3)] )
```

```
mins  <- as.numeric( xl[seq(2, length(xl), 3)] )
secs  <- as.numeric( xl[seq(3, length(xl), 3)] )
```

Eliminate impossible hours or minutes:

```
hsu(hours,60)
```

```
[1]      0      1      2      3      4      5      6      7
 [9]      8      9     10     11     12     13     14     15
[17]     16     17     18     19     20     21     22     23
[25]     24     25     29     33     35     37     41     71
[33]     84     89    302    910    930 150700
```

```
hsu(mins, 60)
```

```
[1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[19] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
[37] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
[55] 54 55 56 57 58 59
```

```
hsu(secs, 60)
```

```
[1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[19] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
[37] 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
[55] 54 55 56 57 58 59
```

```
hours[hours > 23] <- NA
mins[mins > 59] <- NA
hours <- hours + mins/60 + secs/3600 + 12 * pm
```

Subtract 12 from hours over 24, eliminate any left >24:

```
u <- !is.na(hours)
hu <- hours[u]
hu[hu > 24] <- hu[hu >24] - 12
hu[hu >24] <- NA
hours[u] <- hu
```

Correct years to get start time in years units

```
gom$starthours <- hours
gom$years <- gom$years + hours/(24*365)
```

Check all variables:

```
head(gom)
```

```
     Date  Start Interval Duration       years duramin
1 5/2/1970   8:29              4:00 1970.332475    4.00
2 5/2/1970   9:40     1:11    1 1/2 1970.332610    1.50
3 5/2/1970  10:30     0:50    4 1/4 1970.332705    4.25
4 5/2/1970  11:37     1:07        4 1970.332833    4.00
5 5/2/1970 12:58p     1:21    1 3/4 1970.332987    1.75
6 5/2/1970  1:41p     0:43    4 1/2 1970.333069    4.50
      inthours   starthours
1           NA  8.483333333
2 1.1833333333  9.666666667
3 0.8333333333 10.500000000
4 1.1166666667 11.616666667
5 1.3500000000 12.966666667
6 0.7166666667 13.683333333
```

Check that the final days and hour and minute calculations based on the character strings are correct.

9.27 Make and save data frame, omitting missing values;

```
ofd <- na.omit(gom)
dim(ofd)
```

```
[1] 56650     8
```

```
write.csv(ofd, "data/Stephens.csv", row.names=F)
```

10 Functions

```
Grid <- function(xticks, yticks, ylab="",
at=(min(xticks)+ mean(xticks))/2, cex=2.5){
# background for plot using grid of light grey lines

par(mar=c(3,3,6,2))
plot(1, 1,  xlim=range(xticks),  ylim = range(yticks),
       xlab="", ylab="", axes=F, pch="")

# use only interior values of tick ranges in plots
usey <- rep( T, length(yticks) )
usey[c( 1, length(yticks) )] <- F
usex <- rep( T, length(xticks) )
usex[c( 1, length(xticks) )] <- F

# grey lines in both directions
for ( row in yticks[usey] )
lines(range(xticks), c(row, row), col="light grey")
for ( col in xticks[usex] )
lines(c(col, col), range(yticks), col="light grey")
# put ylab on left top, using / to split long expressions
ylabs <- unlist(strsplit(ylab,"/"))

# identify tick marks on both axes
if (length(yticks) > 2)
text(pos=2, rep(min(xticks), length(yticks)-2 ),
   yticks[usey], yticks[usey], cex=2, xpd=T)
if (length(xticks)>2)
text(pos=1, xticks[usex],  rep(min(yticks),
  length(xticks)-2), xticks[usex], cex=2, xpd=T)
lylabs <- min(5, length(ylabs))
if(lylabs > 0)
mtext(ylabs, side=3,line = (5/lylabs)*(lylabs-1):0,
  at = at, cex=cex)

invisible()
}
```

```
condHist <- function(x, y,  xbreaks=0, ybreaks = 0,
         xlab="", ylab="", heightadj=1, widthadj=1){
# counts number of x, y in boxes defined by xbreaks and
ybreaks. A rectangle is displayed over the center of each
box, with area proportional to the count in that box. The
heights and widths of all rectangles may be adjusted by
the heightadj and widthadj.

if(length(x) != length(y))
return(paste(xlab, ylab, "different length"))

# define labels fix na's
use <- !is.na(y) & !is.na(x)
x <- x[use]
y <- y[use]

# specify xbreaks and ybreaks, xmids, and ymids
if (length(xbreaks) > 1) {
 use <- x >= min(xbreaks) & x <= max(xbreaks)
 x <- x[use]
 y <- y[use]
}
if (length(ybreaks) > 1) {
 use <- y >= min(ybreaks) & y <= max(ybreaks)
 x <- x[use]
 y <- y[use]
}

# construct xbreaks and ybreaks if not specified
if (length(xbreaks)==1)
if (xbreaks == 0) xbreaks <- hist(x, plot=F)$breaks
if (length(ybreaks)==1)
if (ybreaks == 0) ybreaks <- hist(y, plot=F)$breaks

if(length(xbreaks)==1)
xbreaks <- min(x) + (0:(xbreaks-1)) *
(max(x)-min(x))/(xbreaks-1)
```

```
if(length(ybreaks)==1)
 ybreaks <- min(y) + (0:(ybreaks-1)) *
 (max(y)-min(y))/(ybreaks-1)
lx <- length(xbreaks) - 1
ly <- length(ybreaks) - 1
if( sum(diff(xbreaks) <= 0) )
 return("xbreaks not increasing")
if( sum(diff(ybreaks) <= 0) )
 return("ybreaks not increasing")

# determine xmid ymid locations for plotting,
xmids <- hist(x, breaks=xbreaks, plot=F)$mids
ymids <- hist(y, breaks=ybreaks, plot=F)$mids
xwidths <- diff(xbreaks)
ywidths <- diff(ybreaks)

# determine counts using hist
xcounts <- hist(x, breaks=xbreaks, plot=F)$counts
ycounts <- hist(y, breaks=ybreaks, plot=F)$counts
bestj <- min( which(ycounts == max(ycounts)) )
colors <- rainbow(ly)[ly - abs((1:ly)-bestj)]

yprop <- matrix(0, ly, lx)
for ( i in 1:lx){
 yi <- y[ x > xbreaks[i] & x <= xbreaks[i+1] ]
 if (length(yi) > 0)
   yprop[, i] <- hist(yi, breaks=ybreaks,
                 plot=F)$counts/xcounts[i]
}

# get maximum proportions relative to widths
maxy <- ywidths
for( j in 1:ly) maxy[j] <-  max(yprop[j,])/ywidths[j]

# adjust the maximum widths and heights to allow some
overlap in rectangles
maxprop <- max(maxy)/ heightadj
xmax <- max(xcounts/xwidths)/ widthadj

# to prevent plots when there are no counts
```

```
yprop[yprop==0] <- NA

# compute rectangle at each point
xleft  <- matrix(xmids - 0.5* xcounts / xmax, lx, ly)
xright <- matrix(xmids + 0.5 * xcounts / xmax, lx, ly)
ybottom <- t(matrix(ymids, ly, lx) - 0.5 * yprop /
maxprop)
ytop <-    t(matrix(ymids, ly, lx) + 0.5 * yprop /
maxprop)

# now graph out all the rectangles specified in the arrays
rect(xleft , ybottom, xright, ytop,
 col=colors[t(matrix(rank(ymids), ly, lx))],
 border=colors[t(matrix(rank(ymids), ly, lx))] )
rect(xbreaks[1] -2* xwidths[1], ymids - 0.5* ywidths,
     xbreaks[1] - xwidths[1], ymids + 0.5*ywidths,
col=colors[rank(ymids)],border=colors[rank(ymids)])

invisible()
}
```