```
#=====================================================================
#  file:          c:\Projects\CBP\Rcourse\ProcessingGroupsRbit007.r
#  function:      processing data by groups
#
#  programmer:    Elgin S. Perry, Ph. D.
#
#  date:           7/4/2014
#
#  address:       2000 Kings Landing Rd.
#                 Huntingtown,  Md.  20639
#
#  voice phone:   (410)535-2949
#  email:         EPERRY@chesapeake.net
#=====================================================================

#install.packages()
#library(lattice) #Used for contour plots [contourplot()]
#library(nlme)    #used for gam Mixed model [gamm()]
#library(MASS)    #used for glm Mixed model [glmmPQL()]
#library(mgcv)    #Wood's gam package
#library(chron)   #date functions
#library(doBy)    # Allows "BY processing similar to SAS
#library(FitAR)   #AR package from McLeod and Zhang
#library(Hmisc)   #stat function by Frank Harrell
#library(cluster) #cluster analysis routines
options(stringsAsFactors = FALSE)
# load libraries
library(chron)
library(mgcv)
# set working directory
ProjRoot <- 'c:/Projects/CBP/Rcourse/'
setwd(ProjRoot);

# load some user defined functions
source("C:/Projects/Rtp/dfsum.r")
source("C:/Projects/Rtp/RTF.r")

# use list() to create a list
contact1 <- list(
  lastname='Jones',
  firstname='Albert',
  phone = list(cell='410-610-2432',land='410-510-3945',fax='410-301-2943'),
  address=c('2742 Long Field Rd.', 'Teetotem', 'Virginia', '22443')
  )
contact1


# methods of referencing list components
contact1[[1]]
contact1$lastname
contact1[['lastname']]
selected <- 'lastname'
contact1[[selected]]
contact1[[2]]
contact1[[3]]
contact1$address

#methods of referencing the components of a list within a list
contact1[[3]][[1]]
contact1$phone[[1]]
contact1$phone$cell
contact1[[3]]$cell

# methods for referencing the elements of vectors in a list
```

```
contact1[[4]][1]
contact1$address[1]
contact1$[4][1]
contact1$[[4]][[1]]
contact1[4][1]

# [] vs. [[ ]] for lists
a <- contact1[1]; class(a)
b <- contact1[[1]]; class(b)
sublist <- contact1[1:2]
sublist
class(sublist)
sublist <- contact1[[1:2]]


# handy functions for examining lists
str(contact1)
names(contact1)
class(contact1)
class(contact1[[1]])
class(contact1$phone)
class(contact1$address)
length(contact1)

# create a 2nd contact
contact2 <- list(
  lastname='Smith',
  firstname='Robert',
  middle = 'A.',
  phone = list(cell='804-225-2352',land='804-540-3945'),
  address=c('3752 Broad Field Rd.','Apartment 3A' ,'Passapatanzy', 'Virginia', '22485')
  )
contacts <- list(contact1,contact2)


# read data as usual
datafile <- paste(ProjRoot,"MAT_5day.csv",sep='');
mat <- read.table(datafile, header=TRUE, sep=",", na.strings="NA", dec=".",
strip.white=TRUE,stringsAsFactors = FALSE)
dfsum(mat)
# [1] "Column Names"
# [1] "StationCode" "StationName" "Season"      "Year"        "Month"       "Date"
# [7] "Time"        "Date.Time"   "Depth"       "Temp"        "Salinity"    "DOpsat"
#[13] "DO"          "pH"          "Turbidity"   "Chlorophyll"

# define a time variable (this uses chron library)
mat$time <- times(paste(mat$Time,'00',sep=':'))
# define a variable to split days in half
mat$time.split <- mat$time < 0.5

# the 'apply' functions

# row and column operations with apply()
num.var <- c( "Temp","Salinity","DOpsat","DO","pH","Turbidity","Chlorophyll")
rowsums <- apply(mat[1:10,num.var],1,sum); rowsums
colmeans <- apply(mat[1:10,num.var],2,mean); colmeans
colmeans <- apply(mat[1:10,10:16],2,mean); colmeans
colmeans <- apply(mat[1:10,],2,mean); colmeans

 fs# grouped data operations with tapply
mn.temp <- tapply(mat$Temp,mat$Date,mean); mn.temp
sd.temp <- tapply(mat$Temp,INDEX=mat$Date,FUN=sd,na.rm=TRUE); sd.temp
# put results in a matrix
mnsd.temp <- cbind(mn.temp,sd.temp)
mnsd.temp
```

```
# put results in a data frame
mnsd.temp <- data.frame(datec=names(mn.temp), mn.temp = mn.temp, sd.temp=sd.temp)
mnsd.temp

# calling tapply() to create a list
mn.temp.list <- tapply(mat$Temp,mat$Date,mean,simplify=FALSE); mn.temp.list
unlist(mn.temp.list)

# using aggregate
mat.daymean <- aggregate(mat[,num.var],list(mat$Date),mean); mat.daymean
class(mat.daymean)

mat$time.split <- as.numeric(mat$time>0.5)  # split the day in two
# compute means for each half day
mat.halfdaymean <- aggregate(mat[,10:16],list(mat$Date,mat$time.split),mean)
mat.halfdaymean

# write you own function for tapply or aggregate
iqr <- function(x)
  {
# x <- mat$Temp
  q25 <- quantile(x,0.25)
  q75 <- quantile(x,0.75,)
  iqr <- q75-q25
  }
mat.temp.dayiqr <- tapply(mat[,'Temp'],list(mat$Date),iqr); mat.temp.dayiqr
mat.dayiqr <- aggregate(mat[,10:16],list(mat$Date),iqr)
mat.dayiqr


# I find it difficult to get useful results using apply functions to apply complex function
such as lm()
turb.reg <- function(dta)
  {
  lm1 <- lm(dta$Turbidity ~ dta$Chlorophyll)
  }
a <- turb.reg(mat)

# make a vector of unique dates
dates <- unique(mat$Date)

# applying functions to elements of a list using sapply or lapply
mat.dpH <- list(mat1 = mat[mat$Date==dates[1],"pH"],
               mat2 = mat[mat$Date==dates[2],"pH"],
               mat3 = mat[mat$Date==dates[3],"pH"])
pH.mean <- sapply(mat.dpH,mean); pH.mean
pH.iqr  <- sapply(mat.dpH,iqr); pH.iqr


mat.reg <- sapply(mat.dl,turb.reg)


# complex analyses of groups using loops.

# define an analysis function
diel.gam <- function(day)
  {
# day <- '3/21/2006'
  # select data for specified date into temporary data frame tdta
  tdta <- mat[day==mat$Date,]
  # put at title for this set of output in rtf file
  RTFtext(paste("Diel Analysis results for",day))
  # create data summary
  num.var <- c( "Temp","Salinity","DOpsat","DO","pH","Turbidity","Chlorophyll")
  mat.halfdaymean <- aggregate(tdta[,num.var],list(tdta$Date,tdta$time.split),mean)
```

```
  mat.halfdaymean
  # call RTFtab() with defaults
  RTFtab(mat.halfdaymean)
  # make table a little nicer
  mat.halfdaymean[,num.var] <- round(mat.halfdaymean[,num.var],2)
  RTFtab(mat.halfdaymean,
         TableTitle=paste('Means of numeric data by half-day for',day),
         vr= c("Group.1","Group.2","Temp","Salinity","DO","pH","Turbidity","Chlorophyll"),
         ch= c("Date","am/pm","Temp","Salinity","DO","pH","Turbidity","Chlorophyll"),
         cw = c(rep(1000,6),1200,1500),
         )
  # fit gam model to selected data
  dogam <- gam(DO ~ s(time,bs='cc'),data=tdta)
  # get predicted values from gam and add to tdta
  tdta$pred <- predict(dogam)
  # plot data, label with day
  plot(DO~time,data=tdta,main=day)
  # overlay predicted line
  lines(pred~time,data=tdta,col='red',lwd=2)
  # get max and min predictions
  range.do <- range(tdta$pred)
  # locate times associated with max and min
  min.pt <- tdta[range.do[1]==tdta$pred,c('time','pred')]
  max.pt <- tdta[range.do[2]==tdta$pred,c('time','pred')]
  min.time <- tdta[range.do[1]==tdta$pred,'Time']
  max.time <- tdta[range.do[2]==tdta$pred,'Time']
  # label max and min on plot
  text(min.pt[1],min.pt[2],'min',cex=1.5,col='red',pos=1)
  text(max.pt[1],max.pt[2],'max',cex=1.5,col='red',pos=3)
  # put plot in RTF file
  RTFput.plt(tmpfile='c:/projects/rtp/TempPng.png')
  # write min/max summary in rtf file
  RTFtext(paste("The maximum DO of", round(max.pt[2],2),"occurred at",max.time))
  RTFtext(paste("The minimum DO of", round(min.pt[2],2),"occurred at",min.time))
  # put a blank line is rtf file
  RTFtext('')
  # write a gam summary anova table in rtf file
  RTFgam.anova(dogam)
  # put the standard r-summary of a gam in the rtf file
  RTFsummary(dogam)
  # put a page break in the rtf file
  RTFpage()

  } #end diel.gam

# call function for each date in mat
daylist <- unique(mat$Date)

# initialize an RTF file
RTFout <- "c:/Projects/CBP/Rcourse/ProcessingGroupsRbit007.rtf"
RTFinit()
# call function for each date in mat using loop and save results to rtf
 for (day in daylist) { diel.gam(day) }
RTFtext('end of file')
RTFclose()
```