```
#======================================================================
#  file:           c:\Projects\CBP\Rcourse\MdCoreTrend\CTstep.r
#  function:       examine core trends data for steps trends that conincide
#                  with methods change, assess with and without adjustment
#
#  programmer:    Elgin S. Perry, Ph. D.
#
#  date:          7/24/2014
#
#  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)
get.ind <- function(x,y)
  { # get index of match for x in y
  ind <- 1:length(y)
  get.ind <- ind[x==y]
  }
vec.strg <- function(x,sep=' ')
  # converts a vector to single string character
  {
  if(length(x) > 1)
    {s <- ""
     for (y in x)
       {s <- paste(s,y,sep=sep)
       }
    }else
    {print('argument not a vector in vec.strg')
     s <- paste(x)
    }
  vec.strg <- s
  } # end of vec.strg

source("C:/Projects/Rtp/dfsum.r")
source("C:/Projects/Rtp/RTF.r")
source("C:/Projects/Rtp/DistFunct.r")
doy <- function(date)
  { # compute day of calendar year for a date
  # date must be of class dates, use dates()
  yr <- years(date)
  fdc <- paste('01/01/',yr)
  fd <- chron(dates = fdc)
  doy <- date-fd+1
  }

#  be sure to change \ to /
ProjRoot <- 'c:/Projects/CBP/Rcourse/MdCoreTrend/'
setwd(ProjRoot);
# file for writing *.rtf results
RTFout <- paste(ProjRoot,"MdCtStep.rtf",sep='')
```

```
# file to temporarily store plots from RTFput.plt()
temp.plot <- paste(ProjRoot,"TempPng.png",sep='')

# check number of fields in data
datafile <- paste(ProjRoot,"CORETrend97_13.csv",sep='');
a <- count.fields(datafile, sep = ",", quote = "\"", skip = 1,
             blank.lines.skip = TRUE, comment.char = "#")
range(a)
#rbind(1:length(a),a)

# read data into dataframe
ct <- read.table(datafile, header=TRUE, sep=",", na.strings="NA", dec=".",
strip.white=TRUE,stringsAsFactors = FALSE)
dfsum(ct)

# [1] "STATION"   "REP_NUM"   "SDEPTH"    "YEAR"      "MONTH"     "DAY"       "TOC_G"
"TSS_G"     "TKNW_G"
#[10] "TP_G"      "TOC"       "TSS"       "TKNW"      "TP"        "NH4"       "NH4_G"
"NO23"      "NO23_G"
#[19] "NO2"       "NO2_G"     "PO4"       "PO4_G"     "DATE"      "LayerCode" "DOC_A"
"DOC_G"     "DOC"
#[28] "NH4_A"     "NO2_A"     "NO23_A"    "PC_A"      "PC_G"      "PC"        "PN_A"
"PN_G"      "PN"
#[37] "PO4_A"     "PP_A"      "PP_G"      "PP"        "TDN_A"     "TDN_G"     "TDN"
"TDP_A"     "TDP_G"
#[46] "TDP"       "TKNW_A"    "TKNType"   "TOC_A"     "TP_A"      "TSS_A"     "TN"
"TN_G"

# transform character date into r-date
ct$date <- as.POSIXct(strptime(ct$DATE, "%d%b%Y"))

# establish the date of lab change
lab.change <- as.POSIXct(strptime("2005-07-01","%Y-%m-%d"))
# create a binary variable to model step change due to lab
ct$step <- as.numeric(ct$date > lab.change)
# create a day of year variable for modeling seasonal effects
ct$doy <- doy(dates(paste(ct$date),format="Y-m-d"))
#  create a year variable for trend
ct$year <- as.numeric(ct$YEAR)

# make a vector of dependent variables for this analysis
deps <- c("TSS","TN","NH4","NO23","NO2","TP","PO4")
# originally had PN and PP in this list, but not PN data before lab change
# log transform the dependent variables
lndeps <- paste('ln',deps,sep='')
for (dep in deps) { ct[,lndeps[get.ind(dep,deps)]] <- log(ct[,dep])}
for (i in 1:length(deps)) { ct[,lndeps[i]] <- log(ct[,deps[i]])}
ct$lnTSS  <- log(ct$TSS )
ct$lnTN   <- log(ct$TN  )
ct$lnNH4  <- log(ct$NH4 )
ct$lnNO23 <- log(ct$NO23)
ct$lnNO2  <- log(ct$NO2 )
ct$lnTP   <- log(ct$TP  )
ct$lnPO4  <- log(ct$PO4 )


# make a vector of stations for this analysis
stats <- unique(ct$STATION)

# look at distributions of dependent variables
for (dep in deps){
  tx <- ct[,dep];
  tx <- tx[!is.na(tx)];
  DistPlot(tx,vlab=dep);
  #readline('hit enter to continue ')
```

```
  }

# look as distributions of log transformed dependent variables
for (dep in lndeps){
  tx <- ct[,dep];
  tx <- tx[!is.na(tx)];
  DistPlot(tx,vlab=dep);
  #readline('hit enter to continue ')
  }

# find the minimum  of all dependents
mvdep <- apply(ct[,deps],MARGIN=2,min,na.rm=TRUE); mvdep

# fix log transform for tss when = 0
ct$lnTSS  <- log(ct$TSS + 0.1)
for (dep in lndeps){
  tx <- ct[,dep];
  tx <- tx[!is.na(tx)];
  DistPlot(tx,vlab=dep);
  #readline('hit enter to continue ')
  }


# create function to analyze data for the dependent at each station
step.test <- function(stat,dep)
  {
#  stat <- stats[8]; dep <- lndeps[1]  # debug line

  ct1 <- ct[ct$STATION==stat,]   # subset to one station of data  (temporary data frame)
  ct1 <- ct1[!is.na(ct1[,dep]),] # eliminate records where dependent missing
  # set up model for analysis
  gamform1 <- as.formula(paste(dep,"~ step + year + s(year,k=8) + s(doy,bs='cc')"))
  # create a gam object based on specified model
  gam1 <- gam(gamform1,data=ct1)
  #gam1 <- gam(lnPP ~ step + year + s(year) + s(doy, bs = "cc"),data=ct1)
  sum.gam <- summary(gam1)
  ct1$pred <- predict(gam1)  # add predicted values to temporary data
  # plot dependent variable versus time
  plot(ct1$date,ct1[,dep],main=paste(dep, 'at',stat),xlab='date',ylab=dep)
  # add predicted values vs. time to plot
  lines(ct1$date,ct1$pred,col='red')
  # put a line on the plot showing point of lab change
  range.dep <- range(ct1[,dep])
  lines(c(lab.change,lab.change),range.dep,col='blue')
  text(lab.change,range.dep[2],'Lab Change')

  # add line to plot showing smooth year trend
  # identify doy with predicted value close to mean for year
  doy.range <- range(ct1$doy)
  doys <- doy.range[1]:doy.range[2]  # create vector from 1st to last doy
  one.year <- data.frame(
    step = rep(0,length(doys)),
    year = rep(unique(ct1$year)[2],length(doys)),
    doy = doys)
  one.year$pred <- predict(gam1,newdata=one.year)
  mn.pred <- mean(one.year$pred)
  one.year$pred.resa <- abs(one.year$pred-mn.pred)
  mn.doy <-  one.year[one.year$pred.resa==min(one.year$pred.resa),'doy']
  ct0 <- ct1
  ct0$doy <- mn.doy
  lines(ct0$date,predict(gam1,newdata=ct0),col='blue')

  #plot(one.year$doy,one.year$pred,type='l')
  #points(med.doy,med.pred,pch=19)
  #points(mn.doy,mn.pred,pch=19,col='red')
```

```
  #lines(c(1,365),rep(mean(one.year$pred),2))

  #  add linear year predictions to plot
  gamform2 <- as.formula(paste(dep,"~ step + year + s(doy,bs='cc')"))
  gam2 <- gam(gamform2,data=ct1)
  ct2 <- ct1
  ct2$doy <- mn.doy
  lines(ct2$date,predict(gam2,newdata=ct2),col='green')
  # put station and dependent at top of plot
  RTFtext(paste("station =",stat,'  dependent =',dep))
  # put the plot in rtf file
  RTFput.plt(height=4,width=6,tmpfile=temp.plot)

  # put an ANOVA table of gam in rtf file
  RTFgam.anova(gam1,main=paste("GAM analysis for station ", stat, "dependent variable", dep))
  # put table of coefficient estimates of gam in rtf file
  RFTgam.coeff.tab(gam1,main=paste("GAM analysis for station ", stat, "dependent variable",
dep))
  # if step is significant, put key work in rtf file
  if (sum.gam$p.pv[2] <= 0.05){RTFtext("step significant")}
  # at end of one station and dep put page break in rtf file
  RTFpage()
  # collect and return the step trend results so can accumulate over stations and dependents
  step.test <- data.frame(station=stat, dep=dep,
step.est=sum.gam$p.coeff[2],step.pv=sum.gam$p.pv[2])

  } #end of step.test

# initialize an empty data frame for accumulating step results
all.step<- data.frame(station = character(0), dep = character(0), step.est = numeric(0),
step.pv <- numeric(0))
str(all.step)

# initialize rtf file
RTFinit("Summary of step trend results for all Core Trend Stations")
RTFtext("Lab Change occurred 07/01/2005")
RTFtext("")
RTFtext("dependents analyzed are:")
depstrng <- vec.strg(deps)
RTFtext(depstrng)
RTFtext("")
RTFtext("Stations analyzed are:")
statstrng <- vec.strg(stats)
RTFtext(statstrng)
RTFtext("")

RTFtext("This analysis implements an intervention model for each dependent variable at each
station.")
RTFtext("In addition to the intervention term, terms for season effect, linear trend and non-
linear trend are included in the model.")
RTFtext("In the figure, the open circles represent the observed data, the red line are
predictions from the full model,")
RTFtext("the blue line represents the nonlinear trend holding season constant, and")
RTFtext("the green line represents linear trend holding season and non-linear trend
constant")
RTFtext("The verticle blue line shows the date of the lab change.")
RTFtext("Following each figure are an ANOVA table and table of fixed coefficients that")
RTFtext("report the significance and direction of the step and linear trends.")

RTFpage()

all.step <- data.frame(station = character(0), dep = character(0), step.est = numeric(0),
step.pv <- numeric(0))
str(all.step)
```

```
start <- Sys.time(); start

statshrt <- stats[1:3]
for (stat in stats)
  {
  stat.step <- data.frame(station = character(0), dep = character(0), step.est = numeric(0),
step.pv <- numeric(0))
  for (dep in lndeps)
    {
    print(paste(get.ind(stat,stats),stat,'    ',get.ind(dep,lndeps),dep))
    one.step <- step.test(stat,dep)
    stat.step <- rbind(stat.step,one.step)
    } # end of dep loop
  stat.tab <- stat.step
  stat.tab$step.est <- round(stat.tab$step.est,6)
  stat.tab$step.pv <- fmt.pval(stat.tab$step.pv)
  RTFtab(stat.tab)
  RTFpage()
  all.step <- rbind(all.step,stat.step)
  } # end of stat loop

finish <- Sys.time(); finish
elapse <- finish - start; elapse
alarm()

all.step$step.sig <- all.step$step.pv <= 0.05
table(all.step$step.sig,all.step$dep)

RTFfreq("dep","step.sig",all.step,pvalue=FALSE)

sigcount <- 53*0.05 + 2 * sqrt(53*0.05*0.95)
RTFtext(paste('Significant counts exceed',round(sigcount,1)))

save(all.step,file=paste(ProjRoot,"AllSteps.rdata",sep=''))

RTFtext('end of file')
RTFclose()
```