```
# to load or include external code from a file
#    source("C:/Projects/Rtp/RTF.r")

#RTFout <- "";  ## set a default output file
RTFinit <- function(text="",fn=RTFout){
  # writes a very simple header for an rtf files
  # defines only one character set = Times New Roman
  # four colors: 1. black, 2. red, 3. green, 4. blue
  write("
{\\rtf1\\ansi\\ansicpg1252\\deff0\\deflang1033\\deflangfe1033{\\fonttbl{\\f0\\froman\\fprq2\\
fcharset0 Times New Roman;}}",fn,sep="\n")
  write("
{\\colortbl;\\red0\\green0\\blue0;\\red255\\green0\\blue0;\\red010\\green120\\blue020;\\red0\\
\green0\\blue255;}",fn,append=TRUE,sep="\n")
  write("\\viewkind4\\uc1\\ ",fn,append=TRUE,sep="\n")
  write("\\pard\\f0\\fs24" ,fn,append=TRUE,sep="\n")
  write(paste(text, "\\par"),fn,append=TRUE,sep="\n")
  rundate <- date()
  write(paste("\\pard",rundate, "\\par"),fn,append=TRUE,sep="\n")
  } # end of RTFinit

RTFclose <- function(fn=RTFout){
  # writes final } of rtf file
  print(fn)
  write("}",fn,append=TRUE)
  } # end of RTFclose

RTFtext <- function(string,fn=RTFout,eol="\n"){
  write(paste("\\pard",string, "\\par"),fn,append=TRUE,sep=eol)
  } # end of RTFtext
RTFput <- function(string,fn=RTFout,eol="\n"){
  write(string,fn,append=TRUE,sep=eol)
  } # end of RTFput

RTFMakRed <- function(string){
  # Append change color red to beginning and change color back to black at end
  # note dependence on color table defined by RTFinit
  string <- paste("\\cf2 ",string,"\\cf1",sep = "");
  } # end of RTFMakRed
RTFMakGreen <- function(string){
  # Append change color green to beginning and change color back to black at end
  # note dependence on color table defined by RTFinit
  string <- paste("\\cf3 ",string,"\\cf1",sep = "");
  } # end of RTFMakGreen
RTFMakBlue <- function(string){
  # Append change color blue to beginning and change color back to black at end
  # note dependence on color table defined by RTFinit
  string <- paste("\\cf4 ",string,"\\cf1",sep = "");
  } # end of RTFMakBlue
RTFMakBold <- function(string){
  # Append bold delimiters
  # note dependence on color table defined by RTFinit
  string <- paste("\\b ",string,"\\b0 ",sep = "");
  } # end of RTFMakBold


RTFpage <- function(fn=RTFout){
  write("\\page",fn,append=TRUE,sep="\n")
  } # end of RTFpage
RTFsummary <- function(robj)
  { # write the output from summary of object to RTF file
  obj.sum <- capture.output(summary(robj))
  RTFtext('')
  for( i in 1:length(obj.sum)) {RTFtext(obj.sum[i])}
```

```
  RTFtext('')
  }  # endo of RTF.summary
RTFprint <- function(robj)
  { # write the output from print of object to RTF file
  obj.sum <- capture.output(print(robj))
  RTFtext('')
  for( i in 1:length(obj.sum)) {RTFtext(obj.sum[i])}
  RTFtext('')
  }  # end of RTF.print

RTFtab <- function(Tabledf,vr=NA,TableTitle=NA,ch=NA,cw=NA,cjs=NA,fn=RTFout){
  # Tabledf <- condf
  # vr = NA          #selected columns / variables
  # TableTitle=NA
  # ch=NA            #column headings
  # cw=NA            #column width
  # fn=RTFout        #output file name
  # if arguments not passed, set some logical defaults
  if (is.na(vr[1])) vr <- names(Tabledf) # default selected columns to all columns of data
frame
  tabcol <- length(vr)                   # set number of columns
  if (is.na(TableTitle)) TableTitle <- "No title given"
  if (is.na(ch[1])) ch <- vr # default column heading to column names
  if (is.na(cw[1])) cw <- rep(1500,times=tabcol) #default column width to 1500 twips
  if (tabcol != length(ch)) {print("number of column headings not equal number of columns")}
  if (tabcol != length(cw)) {print("number of column widths not equal number of columns")}
  cp <- cw
  # set rtf code to put borders on cells
  cellbrdr <- "\\clbrdrt\\brdrs\\brdrw10 \\clbrdrl\\brdrs\\brdrw10 \\clbrdrb\\brdrs\\brdrw10
\\clbrdrr\\brdrs\\brdrw10 "
  # set column heading to bold font
  ch <- RTFMakBold(ch)
  for (i in 2:length(cw)){cp[i] <- cp[i-1] + cw[i]}
  # print(Tabledf)
  # print(paste('tabcol',tabcol))
  # print(vr)
  # print(TableTitle)
  # print(ch)
  # print(cw)
  # print(fn)
  cj <- rep("\\qc",tabcol)  # set column justification for header
  #print(cj);
  # write table title
    write(paste("\\pard\\f0\\fs24 \\par\\par
",TableTitle,"\\par",sep=""),fn,append=TRUE,sep="")
  # put headers on table
  cat("\\trowd\\trrh0",file=fn,append=TRUE,fill=FALSE,sep="")
    for (i in 1:tabcol){

cat(paste("\\clvertalb",cellbrdr,"\\cellx",cp[i],sep=""),file=fn,append=TRUE,fill=FALSE,sep="
")
    }
  cat(paste("\\pard\\intbl",cj[i]),file=fn,append=TRUE,fill=FALSE,sep="")
    for (i in 1:tabcol){
      cat(paste(" ",ch[i],"\\cell",sep=""),file=fn,append=TRUE,fill=FALSE,sep="")
    }
    cat("\\row",file=fn,append=TRUE,fill=FALSE,sep="\n")
  #reset column justification for body of table
  cjd <- cj
  for (j in 1:tabcol){ cjd[j] <- ifelse((class(Tabledf[,vr[j]])=="numeric"),'\\qr','\\ql') }
  if (is.na(cjs[1])) {cj <- cjd} else {cj <- paste('\\q',cjs,sep='')}

  # print(cj);
  # fill rows of table from data frame
  for (i in 1:nrow(Tabledf)) {
```

```
    cat("\\trowd\\trrh0",file=fn,append=TRUE,fill=FALSE,sep="")
      for (j in 1:tabcol){

cat(paste("\\clvertalb",cellbrdr,"\\cellx",cp[j],sep=""),file=fn,append=TRUE,fill=FALSE,sep="
")
      }
    cat("\\pard\\intbl",file=fn,append=TRUE,fill=FALSE,sep="")
      for (j in 1:tabcol){
        #print(paste('row',i,'col',j,vr[j],Tabledf[i,vr[j]],cj[j]))
        cat(paste(cj[j],"
",Tabledf[i,vr[j]],"\\cell",sep=""),file=fn,append=TRUE,fill=FALSE,sep="")
      }
      cat("\\row",file=fn,append=TRUE,fill=FALSE,sep="\n")
    }
  } #end of RTFtab


fmt.pval <- function(pval) {pval <- ifelse((pval<0.0001),"<0.0001",sprintf('%6.4f',pval))}

RTFfreq <- function(aname,bname,data,pvalue=TRUE)
  {
#  data <- mc1; aname <- "S.neg"; bname <- "L.neg"; pvalue=TRUE
  adata <- data[,aname]
  bdata <- data[,bname]
  t <- addmargins(table(adata,bdata))
  dft <- data.frame(t[,1:ncol(t)])
  dft <- cbind(as.character(paste(aname,rownames(t))),dft)
  colnames(dft) <- c(paste(aname,'/',bname),paste(bname,colnames(dft)[2:ncol(dft)]))
  RTFtab(dft,vr=NA,TableTitle=paste('Crosstabulation of ',aname,' and
',bname),ch=NA,cw=NA,fn=RTFout)
  if (pvalue)
    {
    cst <- chisq.test(adata,bdata)
    RTFtext("Pearson's Chi-squared test")
    RTFtext(paste('chi-squared =',round(cst[[1]],4)))
    RTFtext(paste('df =',cst[[2]]))
    RTFtext(paste('p-value =',round(cst[[3]],4)))
    }
  }## end RTFfreq

RTFlm.anova <- function(lmo,main="Analysis of Variance")
  {   # make an rtf anova table from a linear model object
# lmo <- lme1; main <- "Analysis of Variance" #debug
  almo <- data.frame(anova(lmo))
  almo$source <- rownames(almo)
  sst <- sum(almo$Sum.Sq)
  tdf <- sum(almo$Df)
  nr <- nrow(almo)+1
  almo[nr,'source'] <- 'total'
  almo[nr,2] <- sst
  almo[nr,1] <- tdf
  if (sst < 1.0) {almo[,2] <- round(almo[,2],4)}
  if ((1.0<=sst) & (sst<1000)) {almo[,2] <- round(almo[,2],2)}
  if (sst > 1000) {almo[,2] <- round(almo[,2],0)}
  almo[,3] <- round(almo[,3],2)
  almo[,4] <- round(almo[,4],4)
  almo[,5] <- fmt.pval(almo[,5])
  almo
  RTFtab(almo,
         TableTitle = main,
         vr = c('source',"Sum.Sq","Df","Mean.Sq","F.value","Pr..F."),
         ch = c('source','sum of squares','df','mean square','F-stat','p-value'),
         cw = c(2500,1500,1000,1500,1500,1500),
        cjs = c('l','r','r','r','r','r')
        )
```

```
  }## end RTFlm.anova
RFTlm.coeff.tab <- function(lmo,main="Coefficients Table")
  { # tabulate coefficients from linear model fit
#  lmo <- lm6; main="Coefficients Table"
  lm.sum <- summary(lmo)
  lm.coeff <- data.frame(lm.sum$coefficients)
  lm.coeff$source <- rownames(lm.coeff)
  lm.coeff$Estimate <- round(lm.coeff$Estimate,4)
  lm.coeff$Std..Error <- round(lm.coeff$Std..Error,5)
  lm.coeff$t.value <- round(lm.coeff$t.value,4)
  lm.coeff$Pr...t.. <- fmt.pval(lm.coeff$Pr...t..)
  RTFtab(lm.coeff,
         TableTitle = main,
         vr = c("source","Estimate","Std..Error","t.value","Pr...t.."),
         ch = c('parameter','estimate','Std. Err.','t value','p-value'),
         cw =c(4000,1000,1000,1000,1000),
         cjs = c('l','r','r','r','r')
         )
   RTFtext('')
   RTFtext(paste("root mean-square error =",round(lm.sum$sigma,4)))
   RTFtext(paste("r-square =",round(lm.sum$r.squared,4)))
   RTFtext(paste("adjusted r-square =",round(lm.sum$adj.r.squared,4)))
  } # end of RTFlm.coeff.tab

RTFglht <- function(glhto,main)
  {
#  glhto <- DunTrts; main='test'
  glhtsa <- summary(glhto,test=adjusted(type='free'))
  glhtsn <- summary(glhto,test=adjusted(type='none'))
  glht.df <- data.frame(rowlab=names(glhtsa$test$coefficients),
                        estimate=round(glhtsa$test$coefficients,4),
                        s.e. = round(glhtsa$test$sigma,4),
                        t.value = round(glhtsa$test$tstat,4),
                        p.value = fmt.pval(glhtsn$test$pvalues),
                        adj.pval= fmt.pval(glhtsa$test$pvalues))
  #glht.df
  RTFtab(glht.df,
         TableTitle = main,
         vr = c('rowlab','estimate',"s.e.","t.value","p.value","adj.pval"),
         ch = c('Comparison','Difference Estimate','Std.Err.','t-value','p-value',"adjusted
p-value"),
         cw = c(3500,1200,1200,1200,1200,1200),
         cjs = c('l','r','r','r','r','r')
        )
  RTFglht <- glht.df
  } # end of RTFglht

RTFglm.anova <- function(glmo,main="Analysis of Variance")
  {   # make an rtf anova table from a linear model object
# lmo <- glmfit; main <- "Analysis of Variance" #debug
  almo <- data.frame(anova(glmo,test = "Chisq"))
  almo$source <- rownames(almo)
  almo[,2] <- round(almo[,2],4)
  almo[,4] <- round(almo[,4],4)
  almo[,5] <- fmt.pval(almo[,5])
  almo
  RTFtab(almo,
         TableTitle = main,
         vr = c('source',"Df","Deviance","Resid..Df","Resid..Dev","Pr..Chi."),
         ch = c('source','Treatment DF','Deviance','Residual DF','Residual Deviance','p-
value'),
         cw = c(2500,1200,1500,1000,1500,1500),
        cjs = c('l','r','r','r','r','r')
        )
  RTFtext(paste("convergence status = ",glmo$converged))
```

```
  RTFglm.anova <- almo
  }## end RTFlm.anova
RTFcor <- function(cols,dta,labl)
  { #make rtf table of correlations among columns of data frame
#  cols <-
c('CHLA','INT_CHLA',"SOLAR_RAD","TN_NPS_LOAD","TP_NPS_LOAD","NO23_NPS_LOAD","DIP_NPS_LOAD","M
EAN_FLOW"); dta <- jtf; labl <- 'Tidal Fresh'
  ncol <- length(cols)
  cormat <- cor(dta[,cols],use = "pairwise.complete.obs")
  cormat <- round(cormat,4)
  cormatp <- matrix(0,ncol,ncol)
  cormatcp <- matrix('1.00\\par   --',ncol,ncol)
  for (i in 1:(ncol-1))
    {
    for (j in (i+1):ncol)
      {
      #print(paste(i,j))
      CorTest <- cor.test(dta[,cols[i]],dta[,cols[j]],use = "pairwise.complete.obs")
      cormatp[i,j] <- fmt.pval(CorTest$p.value)
      cormatp[j,i] <- cormatp[i,j]
      cormatcp[i,j] <- paste(cormat[i,j],cormatp[i,j],sep='\\par')
      cormatcp[j,i] <- cormatcp[i,j]
      }
    }
  cormatcp <- data.frame(cormatcp)
  names(cormatcp) <- cols
  cormatcp$rownames <- cols
  vars <- c('rownames',cols)
  coljust <- c('l',rep('r',ncol))
  RTFtab(cormatcp,
         TableTitle = paste('Correlation Matrix for',labl),
         vr=vars,
         ch=vars,
         cjs = coljust)
  } #end of RTFcor

# width and height are in inches
.add.png<-function(file,width=3,height=3,verbose=FALSE) {
        # return a hexadecimal version of a file
        max.bytes<-50000000  # maximum file size in bytes (~50MB)
        dat<-readBin(file, what="raw", size=1, signed=TRUE, endian="little",n=max.bytes);
        if(verbose) {
                cat(paste(length(dat),"bytes read\n"))
        }
        paste("
{\\rtf1\\ansi\\deff0{\\pict\\pngblip\\picwgoal",round(width*1440),"\\pichgoal",round(height*1
440)," ",paste(dat,collapse=""),"}}",sep="")
        # paste("
{\\rtf1\\ansi\\deff0{\\pict\\pngblip\\picwgoal",round(width*1440),"\\pichgoal",round(height*1
440)," \n",.chunk.vector(dat),"}}",sep="")
}
RTFput.plt <-
function(tmpfile='c:/projects/rtp/TempPng.png',fn=RTFout,eol="\n",height=6,width=6)
  {
# tmpfile <- 'c:/projects/rtp/TempPng.png'; fn<- RTFout; eol <- "\n"
  savePlot(tmpfile,type='png') #save plot to a file
  plt.string <- .add.png(tmpfile,width=width,height=height,verbose=FALSE)
  write("{\\pard\\fi0\\li0\\f2\\fs20",fn,append=TRUE,sep=eol)
  write(plt.string,fn,append=TRUE,sep=eol)
  write('\\par}',fn,append=TRUE,sep=eol)
  } # end of RTFput.plt

RFTgam.coeff.tab <- function(lmo,main="Coefficients Table")
  { # tabulate coefficients from linear model fit
#  lmo <- lm6; main="Coefficients Table"
```

```
  lm.sum <- summary(lmo)
  lm.coeff <- data.frame(lm.sum$coefficients)
  lm.coeff$source <- rownames(lm.coeff)
  lm.coeff$Estimate <- round(lm.coeff$Estimate,4)
  lm.coeff$Std..Error <- round(lm.coeff$Std..Error,5)
  lm.coeff$t.value <- round(lm.coeff$t.value,4)
  lm.coeff$Pr...t.. <- fmt.pval(lm.coeff$Pr...t..)
  RTFtab(lm.coeff,
         TableTitle = main,
         vr = c("source","Estimate","Std..Error","t.value","Pr...t.."),
         ch = c('parameter','estimate','Std. Err.','t value','p-value'),
         cw =c(4000,1000,1000,1000,1000),
         cjs = c('l','r','r','r','r')
         )
   RTFtext('')
   RTFtext(paste("root mean-square error =",round(lm.sum$sigma,4)))
   RTFtext(paste("r-square =",round(lm.sum$r.squared,4)))
   RTFtext(paste("adjusted r-square =",round(lm.sum$adj.r.squared,4)))
  } # end of RTFlm.coeff.tab

RTFgam.anova <- function(gamo,main="Analysis of Variance")
  {   # make an rtf anova table from a gam model object
#  gamo <- gm1; main <- "Analysis of Variance" #debug
  anov.gamo <- anova(gamo)
#  agamp <- data.frame(anov.gamo$pTerms.table)
#  agamp$type <- ''; agamp$type[1] <- 'parametric terms'
  agams <- data.frame(anov.gamo$s.table)
  agams$type <- ''; agams$type[1] <- 'smoothed terms'
  names(agams)[names(agams)=='edf'] <- 'df' # rename edf
  agams <- agams[,names(agams)!="Ref.df"]   # drop ref.df
#  agamo <- rbind(agamp,agams)
  agamo <- agams
  agamo$source <- rownames(agamo)
  agamo[,1] <- round(agamo[,1],2)
  agamo[,2] <- round(agamo[,2],4)
  agamo[,3] <- fmt.pval(agamo[,3])
  agamo
  RTFtab(agamo,
         TableTitle = main,
         vr = c('type','source',"df","F","p.value"),
         ch = c('Type','Source','edf','F-stat','p-value'),
         cw = c(2000,3000,800,1200,1200),
         cjs = c('l','l','r','r','r')
        )
  }## end RTFgam.anova
RTFbp.tab <- function(bplist,cw=NA)
  { # makes table of statistics in a list of boxplots
#  bplist <- bpall; cw<- NA
  if(exists('bptab')) {remove(bptab)}
  for (i in 1:length(bplist))
    {
    bp.stats <- data.frame(t(bplist[[i]]$stats))
    bp.lbl <- c('lower whisker','lower hinge', 'median', 'upper hinge' ,'upper whisker')
    names(bp.stats) <- bp.lbl
    bp.stats[,"plot group"] <- bplist[[i]]$names
    if (length(bplist) > 1) {bp.stats$label <- bplist[[i]]$label}
    if (length(bplist) > 1) {bp.stats <- bp.stats[,c('label','plot group',bp.lbl)]} else
    {bp.stats <- bp.stats[,c('plot group',bp.lbl)]}
    if (exists('bptab')) { bptab <- rbind(bptab,bp.stats)} else { bptab <- bp.stats}
    }
  if (is.na(cw[1])) { colwid <- rep(1000,ncol(bptab)) } else {colwid <- cw}
  #print(colwid)
  RTFtab(bptab,TableTitle='Statistics for Box Plot',cw=colwid)
  } # end of RTFbp.tab
```

```
#RTFclose()

#  SAS CODE TO BE CONVERTED
# %macro RTFdate(fn,text);
#  see Sys.Date() and Sys.time()
#   filename out "&fn";
#   data null;
#     today = today();
#     put "\pard\f0\fs24 run date = " today date9. "\par";
#   RUN;
# %mend;

#\qc    Centered.
#\ql    Left-aligned (the default).
#\qr    Right-aligned.
```