Examples

Alexander Walker
Alexander.Walker1989@gmail.com

April 28, 2014

1 Installation

The openxlsx package requires an external zip application.

If the command

```
shell("zip")
```

returns "zip is not recognised..." install Rtools from: http://cran.r-project.org/bin/windows/Rtools/ and add the R tools bin directory (defaults to c:\Rtools\bin on windows) to the system path. R tools

2 Examples

2.1 Basic Workbook

```
require(openxlsx)
require(ggplot2)

wb <- createWorkbook()
addWorksheet(wb, sheetName = "Motor Trend Car Road Tests", gridLines = FALSE)
addWorksheet(wb, sheetName = "Iris")
addWorksheet(wb, sheetName = "Conditional Formatting")

## sheet 1
writeDataTable(wb, sheet = 1, x = mtcars,
               colNames = TRUE, rowNames = TRUE,
               tableStyle = "TableStyleMedium9")

setColWidths(wb, sheet = 1, cols = "A", widths = 18)

## write iris data.frame as excel table
writeDataTable(wb, 2, iris, startCol = "L", startRow = 2)

qplot(data=iris, x = Sepal.Length, y= Sepal.Width, colour = Species)
insertPlot(wb, 2, xy=c("B", 16)) ## insert plot

means <- aggregate(x = iris[,-5], by = list(iris$Species), FUN = mean)
vars <- aggregate(x = iris[,-5], by = list(iris$Species), FUN = var)

writeData(wb, 2, means, startCol = "B", startRow=3, borders="rows", borderColour=NULL,
          headerStyle = createStyle(border="TopBottom", fgFill="#FFC7CE", halign="center"))

writeData(wb, 2, vars, startCol = "B", startRow=10, borders="columns", borderColour=NULL,
          headerStyle = createStyle(border="TopBottom", fgFill="#FFC7CE", halign="center"))

setColWidths(wb, 2, cols=2:6, widths = 12) ## width is recycled

## write data with no styling
writeData(wb, 3, x=matrix(rnorm(200000), ncol = 40), startRow=5)

## conditional formating with default style
conditionalFormat(wb, 3, cols=1:50, rows = 4+1:5000, rule="< 0")

## write text to cell B2 and style
titleStyle <- createStyle(fontSize = 24)
writeData(wb, 3, "Normal Numbers", xy = c("B",2))
addStyle(wb, 3, cols = 2, rows=2, style= titleStyle)

saveWorkbook(wb, "basics.xlsx", overwrite = TRUE) ## save to working directory
```

2.2 Stock Price

```
require(openxlsx); require(ggplot2)

wb <- createWorkbook()

## read historical prices from yahoo finance
ticker <- "CBA.AX"
csv.url <- paste("http://ichart.finance.yahoo.com/table.csv?s=",
                 ticker, "&a=01&b=9&c=2009&d=01&e=9&f=2014&g=d&ignore=.csv")
prices <- read.csv(url(csv.url), as.is = TRUE)
prices$Date <- as.Date(prices$Date)
close <- prices$Close
prices$logReturns = c(0, log(close[2:length(close)]/close[1:(length(close)-1)]))

## Create plot of price series and add to worksheet
ggplot(data = prices, aes(as.Date(Date), as.numeric(Close))) +
  geom_line(colour="royalblue2") +
  labs(x = "Date", y = "Price", title = ticker) +
  geom_area(fill = "royalblue1",alpha = 0.3) +
  coord_cartesian(ylim=c(min(prices$Close)-1.5, max(prices$Close)+1.5))

## Add worksheet and write plot to sheet
addWorksheet(wb, sheetName = "CBA")
insertPlot(wb, sheet = 1, xy = c("J", 3))

## Histogram of log returns
ggplot(data = prices, aes(x = logReturns)) + geom_bar(binwidth=0.0025) +
  labs(title = "Histogram of log returns")

## write historical data and  histogram of returns
writeDataTable(wb, sheet = "CBA", x = prices)
insertPlot(wb, sheet = 1, startRow=25, startCol = "J")

## Add conditional formatting to show where logReturn > 0.01 using default style
conditionalFormat(wb, sheet = 1, cols = ncol(prices), rows = 2:(nrow(prices)+1),
                  rule = "> 0.01")

## style log return col as a percentage
logRetStyle <- createStyle(numFmt = "percentage")

addStyle(wb, 1, style = logRetStyle, rows = 2:(nrow(prices)+1),
         cols = "H", gridExpand = TRUE)

setColWidths(wb, sheet=1, cols = c("A", "F", "G", "H"), widths = 15)

## save workbook to working directory
saveWorkbook(wb, "stockPrice.xlsx", overwrite = TRUE)
```

2.3 Image Compression using PCA

```
require(openxlsx)
require(biOps)
require(ggplot2)

## Create workbook and add a worksheet, hide grid lines
wb <- createWorkbook("Einstein")
addWorksheet(wb, "Original Image", gridLines = FALSE)

A <- readJpeg(file.path(path.package("openxlsx"), "einstein.jpg"))
height = dim(A)[[1]]; width = dim(A)[[2]]

## write "Original Image" to cell B2
writeData(wb, 1, "Original Image", xy = c(2,2))

## write Object size to cell B3
## writeData will coerce df to a data.frame,
## if df is a character vector here, colNames is set to FALSE
writeData(wb, 1, sprintf("Image object size: %s bytes",
                         format(object.size(A+0), big.mark=',')),
          xy = c(2,3))  ## equivalent to startCol = 2, startRow = 3

## Plot image
par(mar=rep(0, 4), xpd = NA)
plot(A, bty ="n", frame.plot=F, ann=FALSE)

## insert plot currently showing in plot window
insertPlot(wb, 1, width, height, units="px", startRow= 5, startCol = 2)

## SVD of covariance matrix
rMeans <- rowMeans(A)
rowMeans <- do.call("cbind", lapply(1:ncol(A), function(X) rMeans))
A <- A - rowMeans
C <- A %*% t(A) / (ncol(A) - 1) # covariance matrix of A
E <- svd(C)  ## singlur value decomposition
pve <- data.frame("Eigenvalues" = E$d,
                  "PVE" = E$d/sum(E$d),
                  "Cumulative PVE" = cumsum(E$d/sum(E$d)))

## write eigenvalues to worksheet
addWorksheet(wb, "Principal Component Analysis")
hs <- createStyle(fontColour = "#ffffff", fgFill = "#4F80BD",
                     halign = "CENTER", textDecoration = "Bold",
                     border = "TopBottomLeftRight", borderColour = "#4F81BD")

writeData(wb, 2, x="Proportions of variance explained by Eigenvector" ,startRow = 2)
mergeCells(wb, sheet=2, cols=1:4, rows=2)
```

```
setColWidths(wb, 2, cols = 1:3, widths = c(14, 12, 15))
writeData(wb, 2, x=pve, startRow = 3, startCol = 1, borders="rows", headerStyle=hs)

## Plots
pve <- cbind(pve, "Ind" = 1:nrow(pve))
ggplot(data = pve[1:20,], aes(x = Ind, y = 100*PVE)) +
  geom_bar(stat="identity", position = "dodge") +
  xlab("Principal Component Index") + ylab("Proportion of Variance Explained") +
  geom_line(size = 1, col = "blue") + geom_point(size =3, col = "blue")

## Write plot to worksheet 2
insertPlot(wb, 2, width = 5, height = 4, startCol = "E", startRow = 2)

## Plot of cumulative explained variance
ggplot(data = pve[1:50,], aes(x = Ind, y = 100*Cumulative.PVE)) +
  geom_point(size=2.5) + geom_line(size=1) + xlab("Number of PCs") +
  ylab("Cumulative Proportion of Variance Explained")
insertPlot(wb, 2, width = 5, height = 4, xy= c("M", 2))


## Reconstruct image using increasing number of PCs
nPCs <- c(5, 7, 12, 20, 50, 200)
startRow <- rep(c(2, 24), each = 3)
startCol <- rep(c("B", "H", "N"), 2)

## create a worksheet to save reconstructed images to
addWorksheet(wb, "Reconstructed Images")

for(i in 1:length(nPCs)){

  V <- E$v[, 1:nPCs[i]]
  imgHat <- t(V) %*% A  ## project img data on to PCs
  imgHat.size <- object.size(imgHat)
  V.size <- object.size(V)

  imgHat <- V %*% imgHat + rowMeans  ## reconstruct from PCs and add back row means
  imgHat <- round((imgHat - min(imgHat)) / (max(imgHat) - min(imgHat))*255) # scale
  plot(imagedata(imgHat), bty ="n", frame.plot=F, ann=FALSE)
  imgSize <- V.size + imgHat.size + object.size(rMeans)

  ## write strings to worksheet 3
  writeData(wb, "Reconstructed Images",
            sprintf("Number of principal components used:  %s",
                    nPCs[[i]]), startCol[i], startRow[i])

  writeData(wb, "Reconstructed Images",
            sprintf("Sum of component object sizes: %s bytes",
```

```
                    format(imgSize, big.mark=',')), startCol[i], startRow[i]+1)

  ## write reconstruced image
  insertPlot(wb, "Reconstructed Images", width, height, units="px",
             xy = c(startCol[i], startRow[i]+3))

}

# hide grid lines
showGridLines(wb, sheet = 3, showGridLines = FALSE)

## Make text above images BOLD
boldStyle <- createStyle(textDecoration="BOLD")

## only want to apply style to specified cells (not all combinations of rows & cols)
addStyle(wb, "Reconstructed Images", style=boldStyle,
         rows = c(startRow, startRow+1), cols = rep(startCol, 2),
         gridExpand = FALSE)

## save workbook to working directory
saveWorkbook(wb, "Image dimensionality reduction.xlsx", overwrite = TRUE)
```