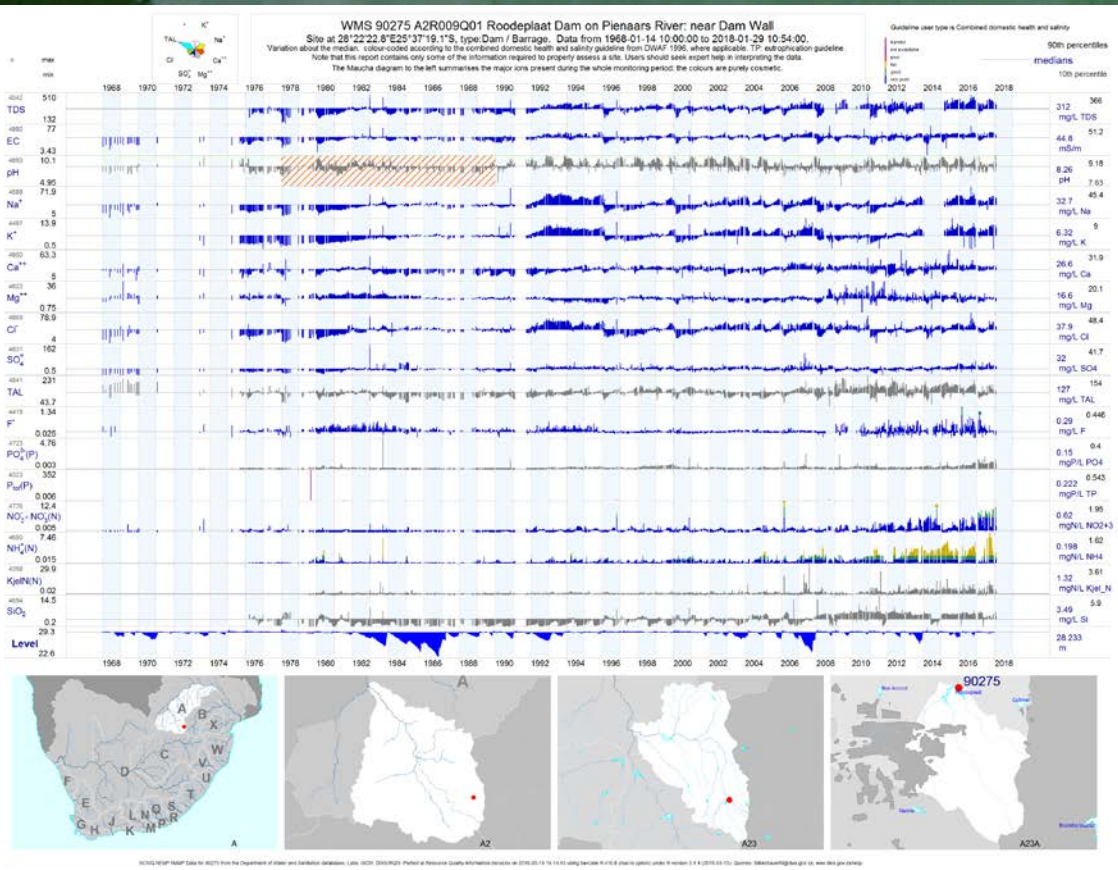
BARCODE time-series plots of inorganic water chemistry variables—RODBC

Resource Quality Information subdirectorate,



water & sanitation

Department:
Water and Sanitation
REPUBLIC OF SOUTH AFRICA

DOCUMENT STATUS

WORKING TITLE:	BARCODE time-series plots of inorganic water chemistry variables—RODBC version of the application that creates BARCODE graphs of the main water chemistry variables
AUTHOR:	Michael Silberbauer
REPORT STATUS:	**Fourth version, Second draft**

RQS REPORT NUMBER: N/0000/00/RD2/2011

DATE: December 2018

Executive summary

The original barcode.aml script developed in the mid-1990s was a response to management requests for a summary of surface water chemistry monitoring activities by the Department of Water Affairs. The script built on the concept developed by Ulrich Nell for the "Yellow Pages" inventories (Swart et al. 1991) and showed water quality results in addition to the monitoring frequency. Since that time, incremental improvements in the software have taken place. In 2005, the plots became more accessible through the web-based inventory of the Water Management System on Google Earth (Silberbauer 2011). By 2011, the ArcInfo-based version of Barcode was becoming obsolete because of the phasing out of the Unix platform for GIS, so the transfer to a PC-based environment began. The first stage of the conversion process was recoding in R, the free statistical and graphics package (R Core Team 2018, Silberbauer 2012, Silberbauer 2017). The next important development was to add the ability to query the main surface water quality database—the WMS—directly, using the RODBC package in R.

The application that creates the Barcode time-series plots, `barcode.R` version 16.9, consists of 20 functions and a main routine. For each site, the functions calculate the ionic strength in milliEquivalents, calculate and construct the geometry of a Maucha ionic diagram, format geographic coordinates as degrees, minutes and seconds, convert strings into a more readable format, calculate and plot the geometry of flow, volume, chemical and physical time series data and plot small locality maps of the monitoring site at four scales.

Introduction

The establishment of an automated inorganic chemistry laboratory at Roodeplaat Dam by the Department of Water Affairs during the 1970s greatly increased the number of sites and the frequency of monitoring that the Department could sustain. The size and complexity of the water quality database meant that individuals no longer knew exactly where data were available, for how long and at what frequency. A programmer at the Hydrological Research Institute (now Resource Quality Information Services), Ulrich Nell, developed programs in Pascal to generate an inventory of all monitoring sites from 1978 to 1988, in what became the "Yellow Pages" series (e.g. Swart 1991). An A4-page summary of each site showed by means of shaded blocks what the sampling frequency was for each of seventeen water quality variables. Ulrich Nell and Willie Geldenhuys produced a set of annual and seasonal box plots for a companion set of reports to summarise four water quality variables at each site, namely total dissolved solids, the sodium absorption ratio, pH and phosphate (e.g. van Veelen et al. 1990). The reports included hand-drafted maps showing the locations of monitoring points.

The philosophy behind barcode.R is that visualisation of information is simpler when everything is on the same page, and the user does not have to flip back and forth through various documents in order to view the relationship between different aspects of the data (see, for example, Tufte 2001). In 1994, the first versions of what later became the barcode program, `multigraph.aml` and `manygraf.aml`, used the ArcInfo 6 Macro Language as a scripting method, simply because it was

the only graphical development platform with multiple user licences available at that time. The script could run in batch mode from the Unix command line and generate summaries for hundreds of stations without manual intervention, so it became a useful tool for researching the data available in a region.

By April 2002, `barcode.aml` could run in the Windows 2000 command-line version of ArcInfo. By 2011, the Unix system had become cumbersome in the mainly Microsoft Windows environment. Porting the system to the R statistics and graphics environment (R Core Team 2018) made sense: R has the advantages of being free, easily available, frequently updated, supported by respected developers, extensible, well documented, programmable and flexible in its data input methods (Rossiter 2010). This document provides details that will help a programmer to maintain and modify the software, or transfer it to another platform if necessary.

Methods

Versions 1 to 9 of the software were coded in Esri Arc Macro Language (AML) as `Barcode.aml`. An example of the code is online (Silberbauer 2002) and discussed by Silberbauer (2009). From version 10 onwards, the software is in R, as `barcode.R`. This document describes version 16.9 of `barcode.R`, which will likely undergo further changes because of enhancements and debugging.

The script comprises 1 main routine and 20 functions. The rest of this section deals with the details of the code.

Main routine

The main routine sets up the database and geographical variables, obtains the site information and sample data for each site and generates the output graphics and text files.

Database settings

The input is from an Informix water quality database, the Water Management System (WMS), accessed through the R package, RODBC (Ripley and Lapsley 2016). For the database connection to function, first set up an ODBC link on the user's computer (Appendix 1). The following commands activate the RODBC package and open the database:

```
library(RODBC)
odbcCloseAll()
channel <- odbcConnect("wmsdb")
wmstables <- sqlTables(channel)
summary(wmstables)
```

The last two lines simply confirm that the correct tables exist in the linked database, and are useful for debugging during an interactive R session.

Constants

The parameters for a batch run are set by assigning values to constants (Table 1).

Table 1. The default parameters in barcode.R

Variable	Default value	Description
plots	TRUE	switch for plotting
type.var	"macro"	select macro or trace elements data set
maps	TRUE	switch for generating locality maps
plots	TRUE	switch for generating time series plots
pdfs	FALSE	switch for choosing output type
year_start	0	start year for subsetting dates - must be 0 (zero) for a full run
year_end	2999	end year for subsetting dates - must be 2999 for a full run
year_update	0	switch for only generating graphs for sites with newer data than year_update-01-01 - must be 0 (zero) for a full run
workdir	"C:\\tmp\\wms\\"	directory for writing output files
logdir	"C:\\data\\WP\\COMPUTER\\website\\"	directory for logging progress
file.meta	paste0 ("C:\\data\\program\\R\\data_description_", type.var, ".txt)	metadata file for describing data
gltype	"DHC"	use a combined domestic health and salinity guideline
type.var	“macro”	type of data to process
PCTile	90	upper percentile shown on each y-axis (can also be 95)
PCTileL	10	lower percentile shown on pH y-axis (can also be 5)

Spatial data

The script invokes three packages for handling spatial data, namely rgdal (Bivand *et al.* 2018), sp (Bivand *et al.* 2013) and maptools (Lewin-Koh and Bivand 2017).

Four locality maps appear along the bottom of each site summary page. The following code reads the required spatial data from shapefiles:

```
if(maps) {
  # Define and read geographical data ------------------------------------------
  shp.dir <- "C:/data_large/av/drainage"
  print (paste("Define and read geographical data from", shp.dir))
  prip <- readOGR(dsn=shp.dir, layer="hca_1geo", stringsAsFactors=FALSE)
```

```
  secp <- readOGR(dsn=shp.dir, layer="hca_2g", stringsAsFactors=FALSE)

  terp <- readOGR(dsn=shp.dir, layer="hca3geo", stringsAsFactors=FALSE)

  qatp <- readOGR(dsn=shp.dir, layer="hca_4", stringsAsFactors=FALSE)

  riv <- readOGR(dsn=shp.dir, layer="wriall500_SimplifyLine_500m",

        GDAL1_integer64_policy=TRUE, stringsAsFactors=FALSE)

  rv <- readOGR(dsn=shp.dir, layer="wriall500", GDAL1_integer64_policy=TRUE,

        stringsAsFactors=FALSE)

  lak <- readOGR(dsn=shp.dir, layer="wla500g", stringsAsFactors=FALSE)

  lak$FEAT_ID <- as.numeric(lak$FEAT_ID)

  town <- readOGR(dsn=shp.dir, layer="smu_500g_simpl_pr_100m",
stringsAsFactors=FALSE)

}
```

The above files and file paths must be accessible on the computer where the script is run.

Layout parameters

A set of constants defines colour coding, range labels and page settings.

Table 2. Page layout parameters in barcode.R

Variable	Value	Description
col.gln	"mediumblue", "seagreen", "gold3", "red", "darkmagenta", "mediumvioletred"	colour coding for each water quality class
comment.gln	"very good", "good", "fair", "poor", "not acceptable", "harmful"	description of each water quality class
gwide	2	line thickness on plots
col.axs	"grey75"	axis colour
col.med	"grey60"	median line colour
col.bdr	"grey75"	border colour
col.patch	"aliceblue"	background patches for year grid
col.edge	"grey88"	vertical lines for year grid
ytop	0.9	fractional height of each plot
ybot	0.25	fractional base of each plot
xleft	0.1	fractional left edge of each plot
xright	0.9	fractional right edge of each plot
pgwidth	11.69	page width in inches (A4)
pgheight	8.27	page height in inches (A4)
np	number of variables + 1	plots per page

Variable	Value	Description
yht	(ytop-ybot)/np	fractional plot height

Timing variables

During program testing, timing variables are helpful for detecting parts of the code that require optimising in order to speed up the process:

```
t0 <- proc.time()[3] # start time (t0)

tn <- proc.time()[3] # time n (tn)
```

or, in the case of map generation time where the biggest bottleneck occurs:

```
t1 <- Sys.time() # start time (t1)

[…]

t2 <- Sys.time() # end time (t2)

tdelta <- as.numeric(t2) - as.numeric(t1) # elapsed time (t2-t1 seconds)
```

Site selection

A normal run of the script will process a list of all available sites from the `nms_wms_geo.dbf` table in the `nms_wms_geo` shapefile generated from the WMS database by another R script, `wms2nms.R`. The input of sites uses a function in the file `monitoringSites.R`.

```
s <- monitoringSites()
```

The next stage is to process selected data for each site.

Data selection

The main loop in the script processes each site in turn, using `nid` as the subscript:

```
for(nid in 1:length(w$PointID)) {
    featid <- w$PointID[nid]
    station <- w$ID[nid]
    name <- w$Name[nid]
    type <- w$Located_On_Type[nid]
```

…

Catchment data come from the inventory quaternary drainage region field, Qat:

```
    qa <- w$Qat[nid]
    pr <- substr(qa,1,1)
    se <- substr(qa,2,2)
    te <- substr(qa,3,3)
```

```
    Ter <- substr(qa,1,3)
    qt <- substr(qa,4,4)
```

A SQL query selects the required water chemistry data from WMS:

```
 q <- paste ("SELECT r.mon_feature_id, r.sample_begin_date, r.sample_begin_time, ",
              " r.sample_begin_depth, r.sample_end_depth, r.preservative_id, ",
              " p.preservative_abbr, i.institution_name, i.institution_abbr, ",
              " r.mon_variable_id, r.result_num_value, r.analysis_meth_id,
                    r.lab_anal_det_limit ",
              " FROM released_result r, preservative p, institution i ",
              " WHERE mon_feature_id = ", featid,
              " AND r.preservative_id = p.preservative_id ",
              " AND r.ana_liaison_id = i.liaisonentity_id ",
              " AND sample_begin_date >= \"", d1, "\" ",
              " AND sample_begin_date <= \"", d2, "\" ",
              " AND mon_variable_id IN (", paste(var.mon.list, var.noc, sep=", "),
")",
              sep="")


  c.in <- sqlQuery(channel, q, stringsAsFactors = FALSE)
```

The `stringsAsFactors` specification ensures that R does not automatically convert strings to the internal "factor" type.

The default date range is the full extent of the dataset, selected by setting `d1` to a very small date and `d2` to a very large date. Otherwise, the values of `d1` and `d2` specify a limited date range. A further option only processes each site if new data are available after a specified date (`year_update`). The reason for this is that monitoring has ceased at many sites, so producing updated plots is pointless unless substantial changes in the plot layout have taken place. This option can save a day or two of processing time.

```
    c.in$date_time <- as.POSIXct(
          paste(c.in$sample_begin_date, c.in$sample_begin_time,
          format="%Y-%m-%d %H:%M:%S")
```

Convert to a wide "spreadsheet" format with the unintuitive melt and dcast functions from reshape2 (Wickham 2007):

```
    c.1 <- c.in[, c("mon_feature_id", "mon_variable_id", "date_time",
```

```
                        "sample_begin_depth", "sample_end_depth",
                        "institution_abbr",

                        "preservative_abbr", "result_num_value")]

    c.1 <- c.1[!duplicated(c.1[, 1:(ncol(c.1) - 1)]), ]

    c.m <- melt(c.1, id=c("mon_feature_id", "mon_variable_id", "date_time",

                          "sample_begin_depth", "sample_end_depth",

                          "institution_abbr", "preservative_abbr"),

                measured=c("result_num_value"))

    c.m <- merge(c.m, vars, by="mon_variable_id")

    c.m <- c.m[order(c.m$date_time, c.m$sample_begin_depth), ]

    c.m$mon_variable_abbr <- TrimFix(c.m$mon_variable_abbr)

    c.m$preservative_abbr <- TrimFix(c.m$preservative_abbr)


    c <- dcast(c.m, mon_feature_id + date_time + sample_begin_depth +
institution_abbr + preservative_abbr ~ mon_variable_abbr, mean)
```

Include NO3_NO2_N_Calc_Water, for those laboratories that provide separate nitrate and nitrite data:

```
if(exists("NO3_NO2_N_Calc_Water", c)) {

  # if the data includes calculated NO2 + NO3,

  if(exists("NO3_NO2_N_Diss_Water", c)) {

       # replace missing Diss values with Calc values if available

       c[which(is.na(c$NO3_NO2_N_Diss_Water)),

       match("NO3_NO2_N_Diss_Water", names(c))]   <-
       c[which(is.na(c$NO3_NO2_N_Diss_Water)), match("NO3_NO2_N_Calc_Water",
       names(c))]

       c <- subset(c, select = -c(NO3_NO2_N_Calc_Water))

        } else { # just rename the Calc to Diss - unlikely scenario

           names(c)[match("NO3_NO2_N_Calc_Water", names(c))] <-

                          "NO3_NO2_N_Diss_Water"

       }

    }
```

File name allocation

Setting all possible file names in one place reduces the possibility of coding errors.

```
if(type.var == "trace") {
     file.png <- paste(wmsdir, "\\", Ter, "_", featid, "_t.png", sep="")
     file.pdf <- paste(wmsdir, "\\", Ter, "_", featid, "_t.pdf", sep="")
     file.ps <- paste(wmsdir, "\\", Ter, "_", featid, "_t.ps", sep="")
     file.csv <- paste(wmsdir, "\\", Ter, "_", featid, "_t.csv", sep="")
     file.zip <- paste(wmsdir, "\\", Ter, "_", featid, "_t.zip", sep="")
   } else {
     file.png <- paste(wmsdir, "\\", Ter, "_", featid, ".png", sep="")
     file.pdf <- paste(wmsdir, "\\", Ter, "_", featid, ".pdf", sep="")
     file.ps <- paste(wmsdir, "\\", Ter, "_", featid, ".ps", sep="")
     file.csv <- paste(wmsdir, "\\", Ter, "_", featid, ".csv", sep="")
     file.zip <- paste(wmsdir, "\\", Ter, "_", featid, ".zip", sep="")
   }
```

Text file

Advanced users of the water quality data may prefer to make their own time series plots, so the script exports zipped comma-separated-value (CSV) files. The current default is `pkzip` – if it is not available, R's `zip` or another alternative compression program may work.

```
if (file.exists(file.csv)) shell(paste("del /f", file.csv))
 if (file.exists(file.zip)) shell(paste("del /f", file.zip))
 cz <- c # make a data frame for output
 cz$Station <- station
 for(npr in 1:length(cz$Preserve))
       cz$Preserve[npr]=preservative(cz$Preserve[npr])
 cz$Qat<-qa
 write.csv(cz, file.csv, na="#n/a", row.names=FALSE)
 MetaFile(type.var, barcode_version, var.mon.list,
                 paste0("site ", featid, " (", station, ")", " by:\n ",
c.source.name, ". (", c.source.abbr, ")" ),
                 name,
                 file.meta
        )

 # run Pkzip to compress the file and add the metadata file
 shell(paste("pkzipc -add -silent=banner -move", file.zip, file.csv))
 shell(paste("pkzipc -add -silent=banner", file.zip, file.meta))
```

Time-series plots

The original software in Arc/Info created compact, flattened PDF files. R creates full vector PDF files, which are very large, so the standard output for the R version is PNG. Plotting uses the R `fig()` method to arrange multiple plots on the same page. Where no data are available during the selected period the script produces an empty plot.

Where data are available for a constituent, the script calculates the minimum, median, percentile (Table 1) and maximum. The command `plot()` with `type='n'` sets up the plot extent without writing anything to the graphics page. This allows the layering of graphics, for example creating a base layer of shaded patches to highlight alternate months (Figure 1).

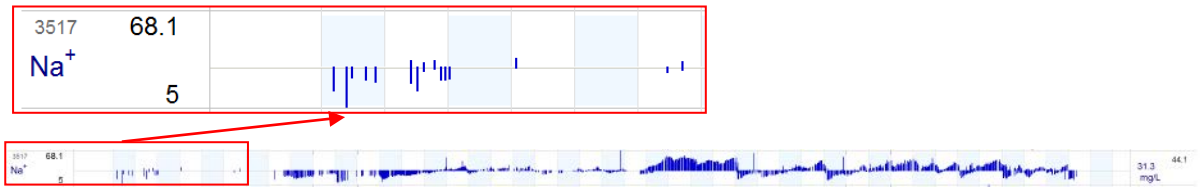


Figure 1. Enlarged view of a portion of a time series plot of sodium at Roodeplaat Dam from 1968 to 2011.

Plots consist of bars showing deviation from the long-term median value. This structure emphasises seasonal and episodic trends (Figure 2). Colour-striping of each bar helps the user gauge what the values mean in terms of the water quality guidelines (DWAF 1996). The striping method only works as expected for values above the median line, so those below the median are coded with a single colour rather than stripes to avoid giving misleading information (Figure 2).

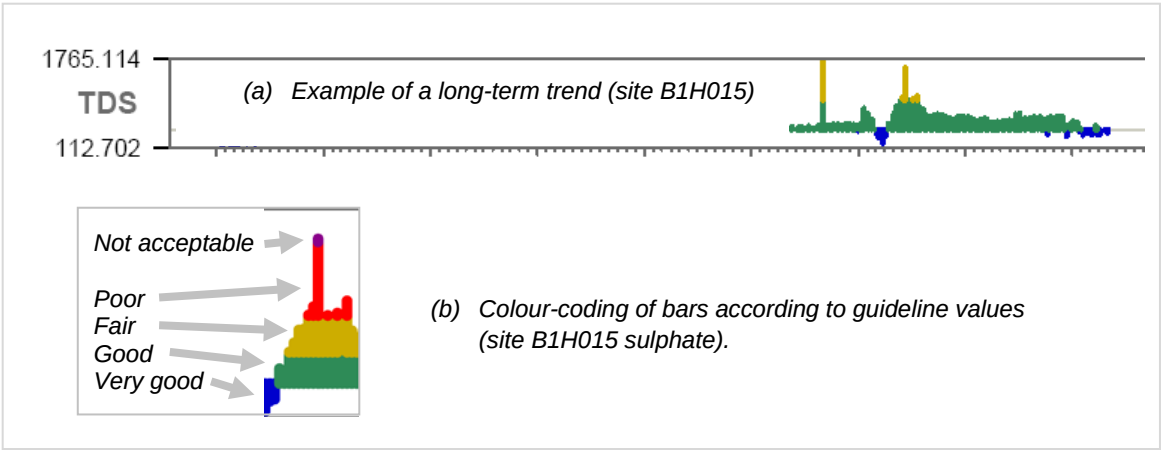


Figure 2. Details of (a) barcode plot showing the TDS trend for an eight-year period and (b) guideline colour-coding of bars.

Through the years, various users of the data have remarked on a discrepancy in the pH values between 1979 and 1989, apparently because of an acid wash used in the automated analyses at that

time (Ramjukadh *et al.* 2017, 2018). The period of uncertainty is highlighted on the output and in the metadata (Figure 3).

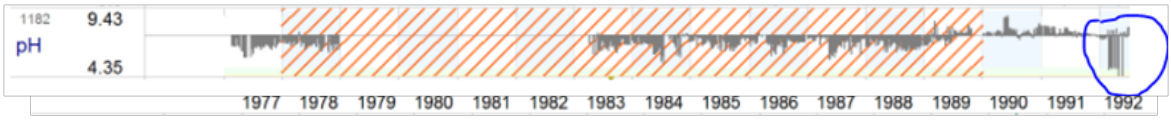


Figure 3. Enlarged view of the 1977-1992 section of a time series plot of pH at Misverstand Dam, G1R003, with the period of uncertain pH data hatched in red. Note that the pH in 1977 and 1992 is also low. Whether the latter is a real phenomenon is unclear.

Functions

Version 16.9 of `barcode.R` consists of 20 functions and a main routine. For each site, the functions calculate the ionic strength in milliEquivalents, calculate and construct the geometry of the Maucha diagram, format geographic coordinates as degrees, minutes and seconds, convert strings into a more readable format, calculate and plot the geometry of flow, volume, chemical and physical time series data and plot small locality maps of the monitoring site at four scales, for the primary, secondary, tertiary and quaternary drainage regions.

chemform

`chemform` formats the labels for each variable, taking care of superscripts and subscripts using the R `expression()` function, e.g. $SO_4^{=}$.

cname

The WMS variable field names are too long for annotation, and `cname` simply abbreviates the name. In some cases, `cname` uses a more common abbreviation, e.g. DMS_Tot_Water becomes TDS rather than DMS.

degs

`degs` is a service function that uses the R sp function `dms` to format a coordinate in decimal degrees as degrees, minutes and seconds. For example, -28.73556 becomes 28°44'08.0" S.

flowdata

`flowdata` retrieves a time series for flow in m^3/s. `flowdata` can read from a text file exported from Hydstra, or from the online hydrological database.

guideline

Each water user type has a different set of guideline variables and values. The `guideline` function defines the range cut-off levels for each user type.

locmapplot

`locmapplot` uses the R packages `mapplot` and `sp` to create a hierarchical set of four small maps showing the location of the monitoring point. The maps are at primary, secondary, tertiary and quaternary drainage region level.

Maucha

The `Maucha` function applies the algorithm of Maucha (1932) as modified by Broch and Yake (1969), implemented in Pascal by Silberbauer and King (1991) and coded in R by Susanne Walford (Lakehead University, Canada—*pers. comm.*). The Maucha diagram summarises the chemical nature of water based on the major dissolved anions and cations (**Figure 4**).

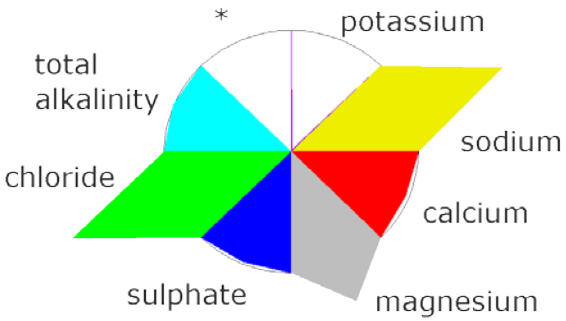


Figure 4. Maucha diagram (Maucha, 1932). The total area of the coloured kite shapes representing anions on the left balances the area of the cations on the right. The original diagram has separate rays for bicarbonate and carbonate (*), which are collapsed into total alkalinity in this version because of uncertainties in calculating the values using pH.

MauchaData

The `MauchaData` function converts the input chemical data in mg/L to ionic concentrations, milliequivalents per litre, abbreviated as mEq/L (Table 3).

Table 3. Factors for converting from mg/L to mEq/L.

Ion	Conversion from WMS value in mg/L
K^+	K_Diss_Water / 39.0983
Na^+	Na_Diss_Water / 22.98977
Ca^{2+}	2 * (Ca_Diss_Water / 40.08)
Mg^{2+}	2 * (Mg_Diss_Water / 24.305)
Cl^-	Cl_Diss_Water / 35.453
SO_4^{2-}	2 * (SO4_Diss_Water / (32.06 + (4 * 15.9994)))
TAL (as HCO_3^-)	TAL_Diss_Water / (1.0079 + 12.011 + (3 * 15.9994))

MauchaKey

The `MauchaKey` function places a labelled Maucha symbol at a specified position to serve as an explanatory key to the ions.

MetaFile

`MetaFile` uses the descriptive information available on the WMS database to list periods that analytical methods were in effect and provide information about detection limits in effect throughout the data record for each constituent. `MetaFile` also gives contact details and a suggested citation format.

patches

In order to help the viewer's eye follow the time axis from one plot to the next, the `patches` function shades alternate time periods in light blue. Patches are one month wide for plots of less than 367 days and one year wide otherwise. Some refinement of colour and intensity is may still be necessary for optimal display on monitors and printed pages (Figure 5).

pHplot

The presentation of pH data is tricky, because the desirable value is 7 (neutral), so extreme deviations above or below 7 suggest water quality impairment. For this reason, `barcode.R` plots pH data as bars around a baseline of 7 rather than the median. The candy-striped bars do not work for this layout, so horizontal patches mark undesirable values (Figure 5).

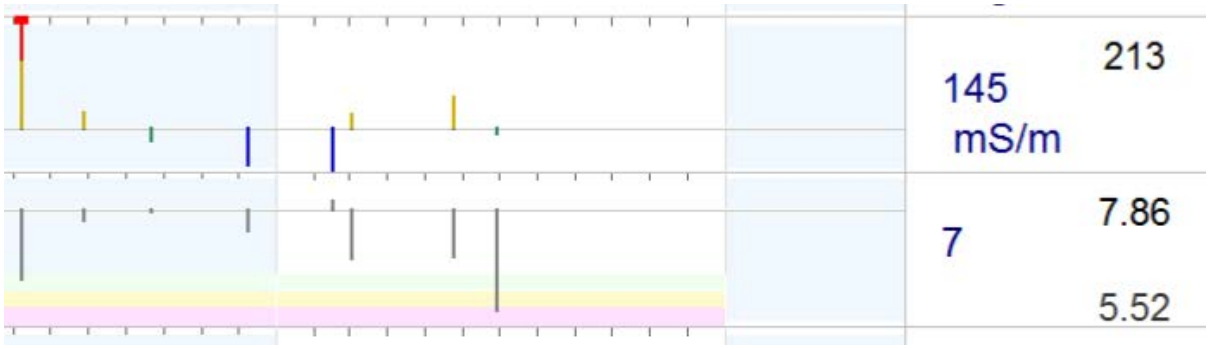


Figure 5 Section of time series chart of site 185085, showing vertical time-period patches and horizontal guideline patches. The top row is conductivity, where colour-coded data bars start at the median value, while the bottom row is pH, where the data bars are grey and the background is colour-coded.

preservative

The WMS database represents sample preservative methods as single-character codes, which the function `preservative` expands to the full description (Table 4).

Table 4. Preservative codes used in WMS.

Code	Preservative
0	None or unknown
1	HgCl2
2	HNO3
3	H2SO4
5	Cool
6	Lugol
7	Formalin

Code	Preservative
9	L-Ascorbic Acid
?	Na2SO4
E	Filter, HNO3, 4°C
F	NaOH, 2°C < 4°C
M	Umgeni Water
Z	Not applicable

TrimFix

Text fields in the WMS database may contain spurious blanks and special characters that sometimes interfere with queries and string comparisons. This function trims leading and trailing blanks and replaces unwanted characters with underscores ("_"). Partially superseded by `str_trim()` from the `stringr` package.

type.glnlabel

`type.glnlabel` assigns the long description for each 3-letter user type (Table 1).

units

`units` defines the unit abbreviation for each variable, in most cases mg/L. One could implement more complex expressions, e.g. $mg.L^{-1}$, but these are sometimes less legible when displayed or printed at a low resolution.

VariableList

`VariableList` selects the variables to report for each type of barcode report, most commonly "macro" for inorganic chemistry.

(vnm)

(`vnm`, not currently used, expands an abbreviated variable name to a long variable name, e.g. EC to electrical conductivity.)

voldata

`voldata` retrieves a time series for dam supply level, where applicable. `voldata` can read from a text file exported from the hydrological database, Hydstra, or from the online hydrological database website www.dwa.gov.za/hydrology, when it is available.

wmscheck

`wmscheck` reports a SQL query error. Then crashes anyway.

Discussion and conclusion

The simple program structure of `barcode.R` reflects its origins in Arc Macro Language. A skilled R programmer may therefore find areas for code optimisation. The slowest process is map generation, taking approximately 8 seconds per site.

The use of PNG bitmap files as output instead of PDF vector files as in the original AML version is a matter of expediency. The old PDF files created with ArcInfo and Adobe Acrobat were 4% of the size of those created by R, because of flattening of map data. Small file size is an important consideration when presenting graphics on the Internet, and the PNG files are 8% of the size of the most compact file obtained by processing an R Postscript file with Adobe Acrobat Distiller. The clarity and scalability of vector graphics in PDF files are far superior to those of bitmaps such as PNG, so the use of compact vector output is worth investigating in future.

Direct querying of the WMS database through `RODBC` was introduced in version 12.0. It is more efficient than exporting intermediate text files and involves fewer processing steps.

The information in the time-series graphs is often densely packed, and design improvements could include horizontal white space between the plots and light grey shading for the background patches. Only samples that pass quality control checks such as cation-anion balance should appear as colour-coded bars—the rest should be dark grey.

Finally, software support documents like this one are easier to maintain using version control and the markdown approach, so RQI should investigate this for future versions.

Acknowledgements

Willie Geldenhuys and Deon van Zyl have assisted with the formulation of SQL queries. The Resource Quality Information team in the Resource Quality Information Services directorate have, through the decades, performed an enormous amount of work to establish and maintain the WMS: Bets Davies, Triana Louw, Elna Vermaak, Collen Dlamini, Marica Erasmus and Nelisiwe Mkhaliphi have been particularly supportive. Finally, the water quality database could not exist without the efforts of many samplers, sample receivers and laboratory analysts since the 1960s.

References

Broch, E. S. and Yake, W. (1969). A modification of Maucha's ionic diagram to include ionic concentrations. Limnology and Oceanography, 14:933-935.

DWAF (1996). Water quality guidelines volume 1: Domestic water use. Second edition. Department of Water Affairs and Forestry, Pretoria.

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/ .

Lewin-Koh, N. J. and Bivand, R. 2017. maptools: Tools for reading and handling spatial objects. R package version 0.9-2. http://cran.r-project.org/web/packages/maptools/

Maucha, R. (1932). Hydrochemische Methoden in der Limnologie XII, pages 1-173. Schweizerbart, Stuttgart.

Bivand, R. S., Pebesma, E. J., and Gomez-Rubio, V. (2013). Applied spatial data analysis with R. Springer, NY. http://www.asdar-book.org/

Ramjukadh, C-L., Silberbauer, M. J. and Taljaard, S. (2017). A discontinuity in pH values in the South African national water quality monitoring database. Southern African Society of Aquatic Scientists congress 25-29 June 2017. http://www.dwa.gov.za/iwqs/water_quality/NCMP/nwrqsr.aspx

Ramjukadh, C.-L., Silberbauer, M., and Taljaard, S. (2018). An anomaly in pH data in South Africa's national water quality monitoring database – implications for future use. *Water SA*, 44(4), 760–763. http://dx.doi.org/10.4314/wsa.v44i4.23

Ripley, B. and Lapsley, M. (2016). RODBC: ODBC Database Access. R package version 1.3-13. http://CRAN.R-project.org/package=RODBC

Rossiter, D. G. (2010). Introduction to the R Project for Statistical Computing for use at ITC. International Institute for Geo-information Science & Earth Observation, Enschede, the Netherlands.

Silberbauer, M. J. (2002). Barcode.aml
http://www.dwa.gov.za/iwqs/water_quality/NCMP/av/barcode.pdf

Silberbauer, M. J. (2009). Methods for visualising complex water quality data. PhD thesis, University of Cape Town, Rondebosch, Cape Town. http://www.riv.co.za/wv/

Silberbauer, M. J. (2012). BARCODE time-series plots of inorganic water chemistry variables—R version (Conversion to R of the application that creates BARCODE graphs of the main water chemistry variables) (version 1). RQS report number: N/0000/00/RD2/2011.

Silberbauer, M. J. (2017). Internet based data sharing solutions for facilitating water quality data distribution to researchers, stakeholders, and policy makers, using R, Google Earth and leaflet. IAHS Scientific Assembly, 2017: IAHS 2017-327.
http://www.riv.co.za/cv/pdf/Innovative_ICT_tools_for_water_management_and_science_3.pdf and

http://www.riv.co.za/cv/pdf/Internet_based_data_sharing_solutions_for_facilitating_water_quality_data.pdf

Silberbauer, M. J. (2018). KML-based online spatial inventory of water chemistry data—R version. RQS report number: N/0000/00/RD1/2011, Resource Quality Services, Department of Water Affairs, Pretoria.

Silberbauer, M. J. and King, J. M. (1991). Geographical trends in the water chemistry of wetlands in the south-western Cape Province, South Africa. South African Journal of Aquatic Science, 17(1/2):82-88.

Swart, S. J., van Veelen, M., and Nell, U. (1991). Water quality data inventory, volume 1: Drainage regions A, B, C, D, E, F, TR146. Technical report, Hydrological Research Institute, Department of Water Affairs, Pretoria.

Tufte, E. R. 2001. The visual display of quantitative information. Graphics Press, Cheshire, Connecticut, second edition.

van Veelen, M., Nell, U., and Geldenhuys, W. F. (1990). Surface water quality of South Africa 1979-1988. Volume 2: Drainage Region A and B. Technical report TR145, Hydrological Research Institute, Department of Water Affairs, Pretoria.

Wickham, H. (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/

Appendix 1

Implementing the Informix database server at RQIS

Changes to user's PC

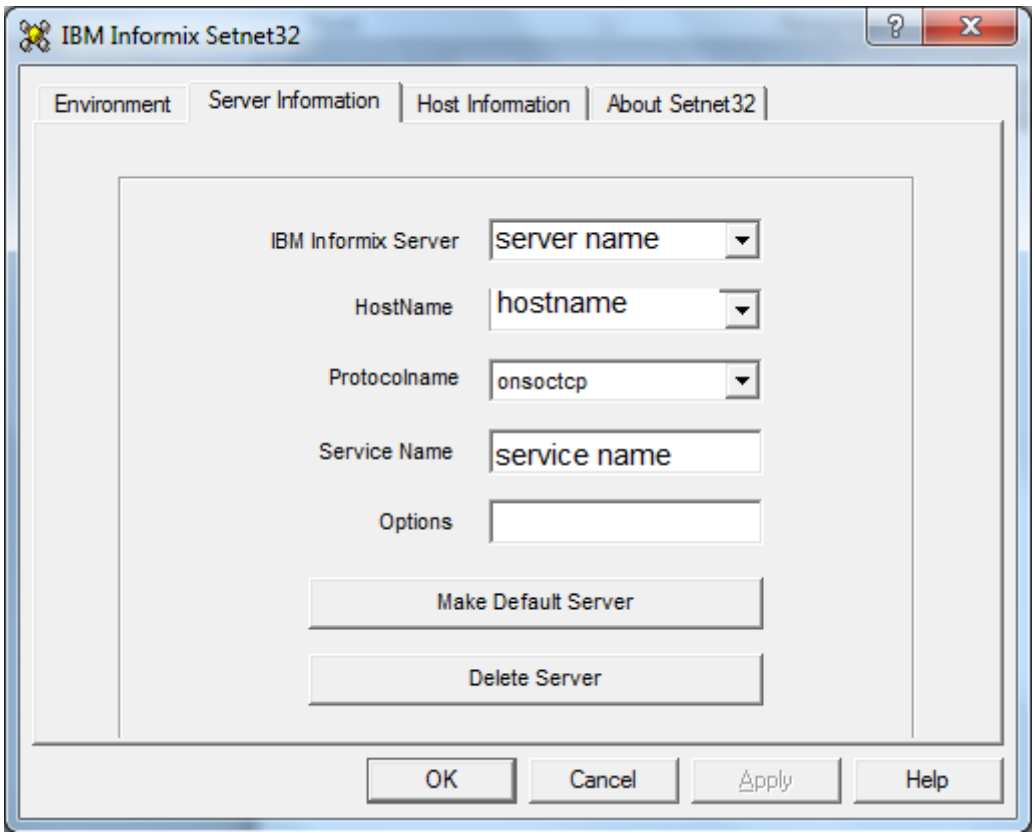
Note that the version of the software is dependent on the user's PC operating system and the latest version of ICONNECT that is available. Consult the IT manager for the current names of servers and hosts.

Install
`I:\WMS-Setup\ICONNECT_2.70\WINDOWS\Informix-Connect.msi`

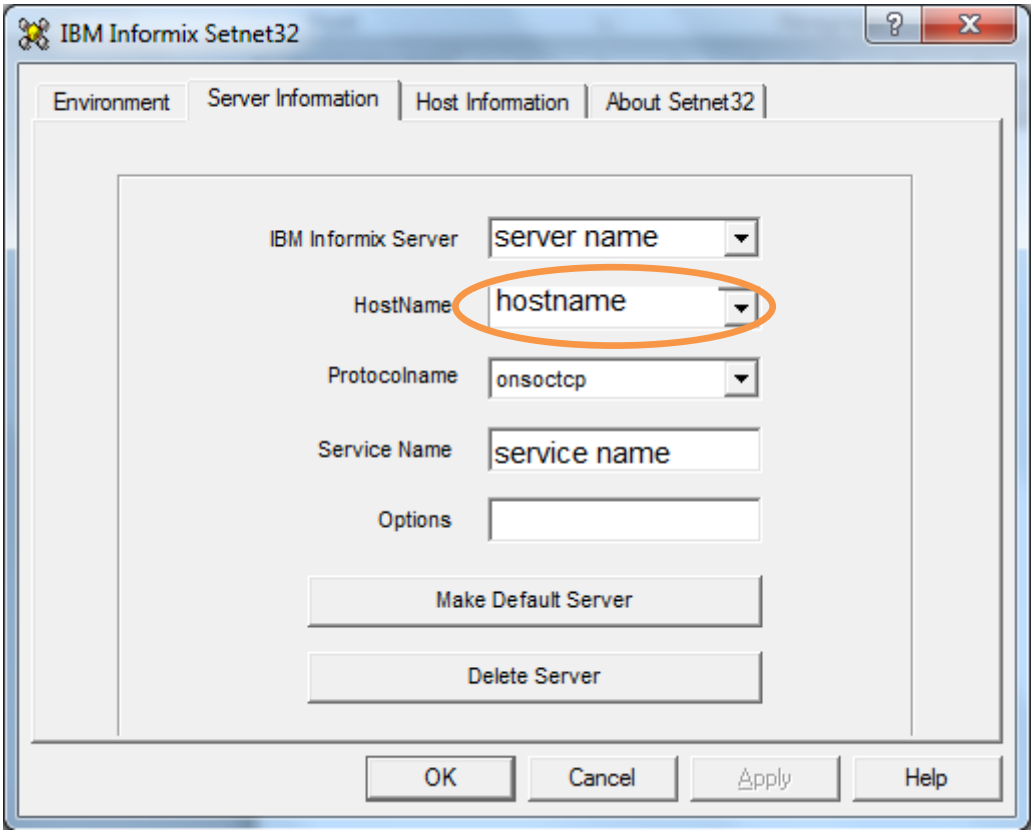
Open Informix Setnet32:
Start – All Programs – Informix 2.7 – Setnet32

Click on Server Information Tab.

Locate the record where Informix Server = **`[current server name]`**

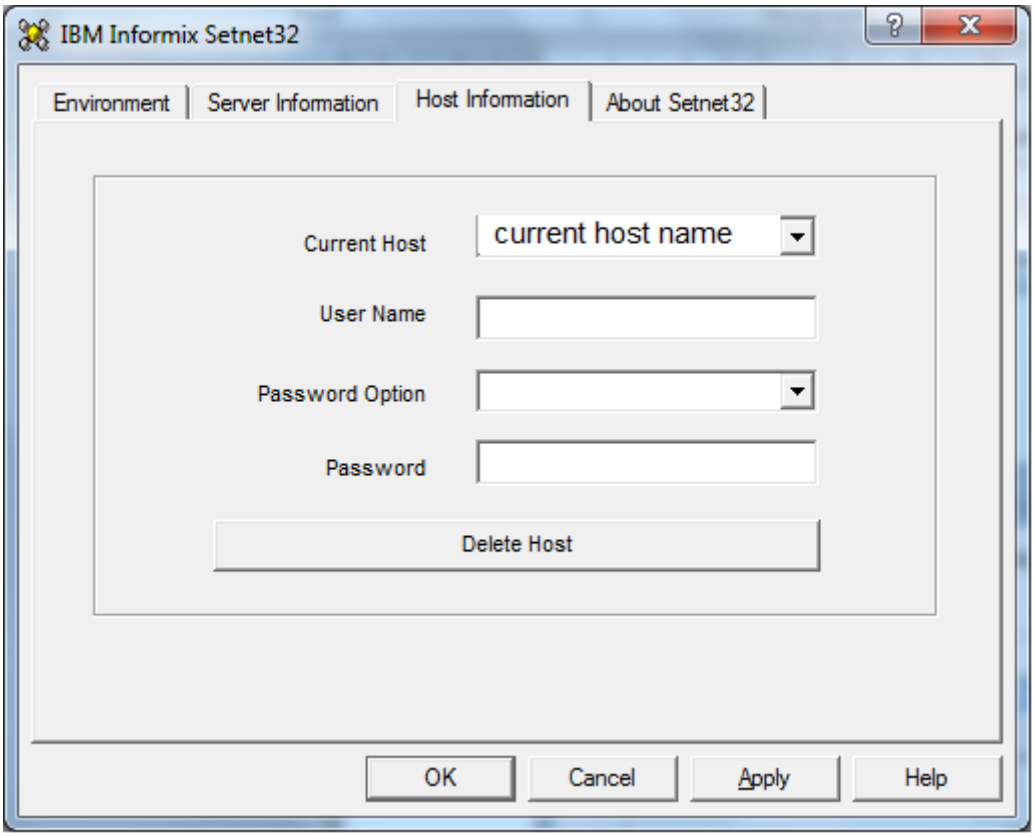


Change Hostname to `[current host name]` (or current IP address)



Click Apply

Click Host Information Tab

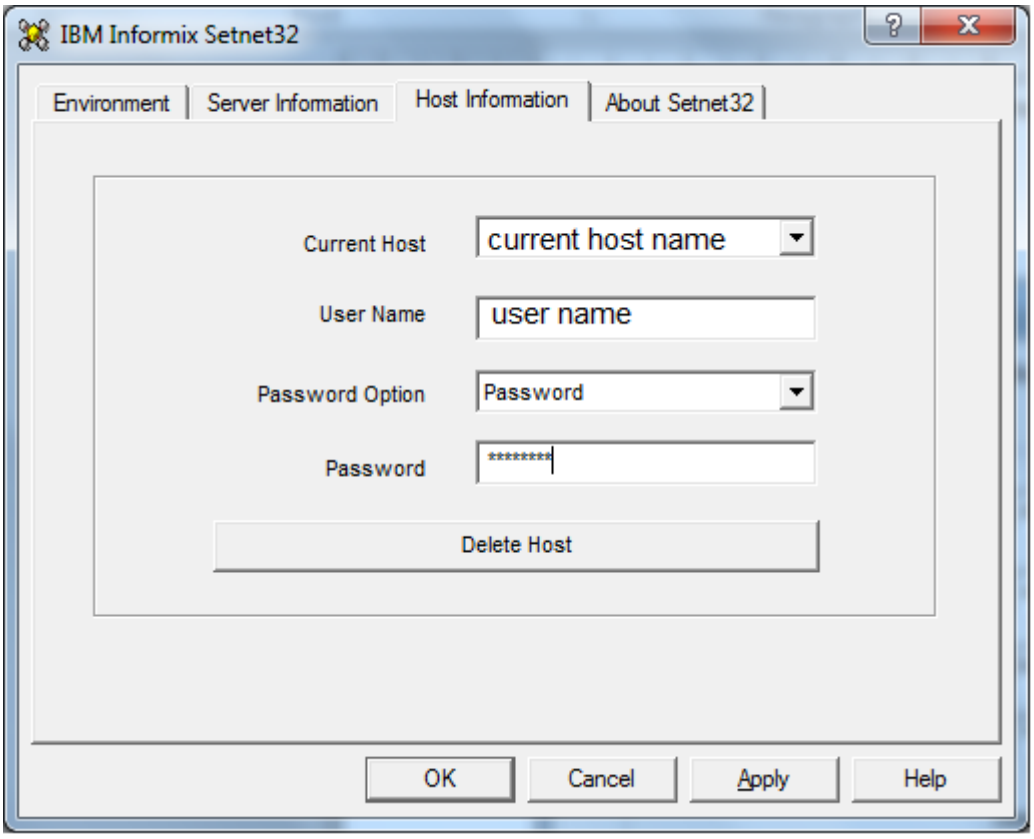


Enter:

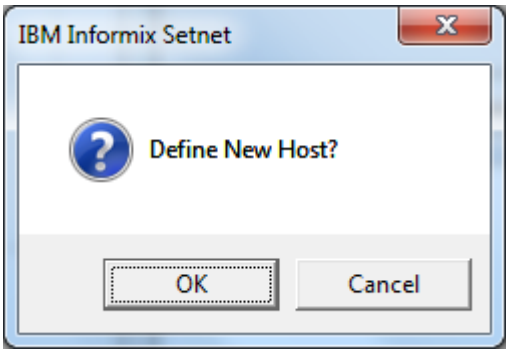
Username = ***** (Your WMS username)

Password Option = Password

Enter password obtained from IT manager.



Click Apply



Click OK

Click OK to close InformixSetnet32

Test if WMS application connects to database.

You may need to copy the **`services`** entry from **`I:\WMS-Setup\SERVICES`** to the local PC – consult the IT manager for advice.

Appendix 2

Uploading the output of barcode.R to the Internet

The drive mapping and upload method is system-specific. Furthermore, certain web files need to be in place for users to find the barcode files online (Silberbauer 2018).

The upload files are stored in a batch file, **`robocopy_ncmp.bat`**:

```
rem Q:\ is the mapping to the web server drive
rem uploads are logged in c2qkmlncmp.log
rem Run as a batch file with %%I - for interactive use, change %%I to %I

rem compress all related R scripts into a single file
pkzipc -add=update C:\tmp\wms\kml_WMS_mnpts_html_R.zip c:\data\program\R\barcode.R
c:\data\program\R\monitoringSites.R c:\data\program\R\toTitle.R
c:\data\program\R\NCMP_miniplots.R c:\data\program\R\Maucha_per_site_chubby.R
c:\data\program\R\kml_WMS_mnpts_html.R  c:\data\program\R\wms2nms.R

rem generate a log file name based on date and time
for /f "tokens=2 delims==" %I in ('wmic os get localdatetime /format:list') do set
datetime=%I
rem batch version of script:
rem for /f "tokens=2 delims==" %%I in ('wmic os get localdatetime /format:list')
do set datetime=%%I

set logfile=C:\data\WP\COMPUTER\website\c2qncmp_%datetime:~0,8%-
%datetime:~8,6%.log
echo %logfile%

rem upload all barcode ZIP output
robocopy /MAXAGE:20181101 /s /xo /v /np /log+:%logfile% C:\tmp\wms\ Q:\wms\data\
*.zip

rem upload all barcode PNG images
robocopy /MAXAGE:20181101 /s /xo /v /np /log+:%logfile% C:\tmp\wms\ Q:\wms\data\
*.png

rem upload any barcode JPG images
robocopy /MAXAGE:20181101 /s /xo /v /np /log+:%logfile% C:\tmp\wms\ Q:\wms\data\
*.jpg

for /f "tokens=2 delims==" %I in ('wmic os get localdatetime /format:list') do set
datetime=%I

rem popup messages when job is complete
msg "%username%" %datetime:~0,8%-%datetime:~8,6% robocopy_ncmp.bat completed
rem javascript version
mshta javascript:alert("NCMP web update finished");close();
```