Processing and organizing SWMP time series for analysis

Marcus W. Beck[1] Todd D. O'Brien[2]

[1]ORISE, USEPA NHEERL Gulf Ecology Division
Email: beck.marcus@epa.gov

[2]NOAA/NMFS COPEPOD Project
Email: todd.obrien@noaa.gov

Objectives and agenda

- Objectives
  - How can SWMP data quality be evaluated and handled?
  - How can data be selected and removed to facilitate analysis?
  - What are some ways that data are combined and why would this be done?

Objectives and agenda

- Objectives
  - How can SWMP data quality be evaluated and handled?
  - How can data be selected and removed to facilitate analysis?
  - What are some ways that data are combined and why would this be done?
- Agenda
  - Handling QAQC flags
  - Appropriate use of data subsets
  - Combining data for comparisons

Interactive portion

You can follow along in this module:

- dataset2
- script2

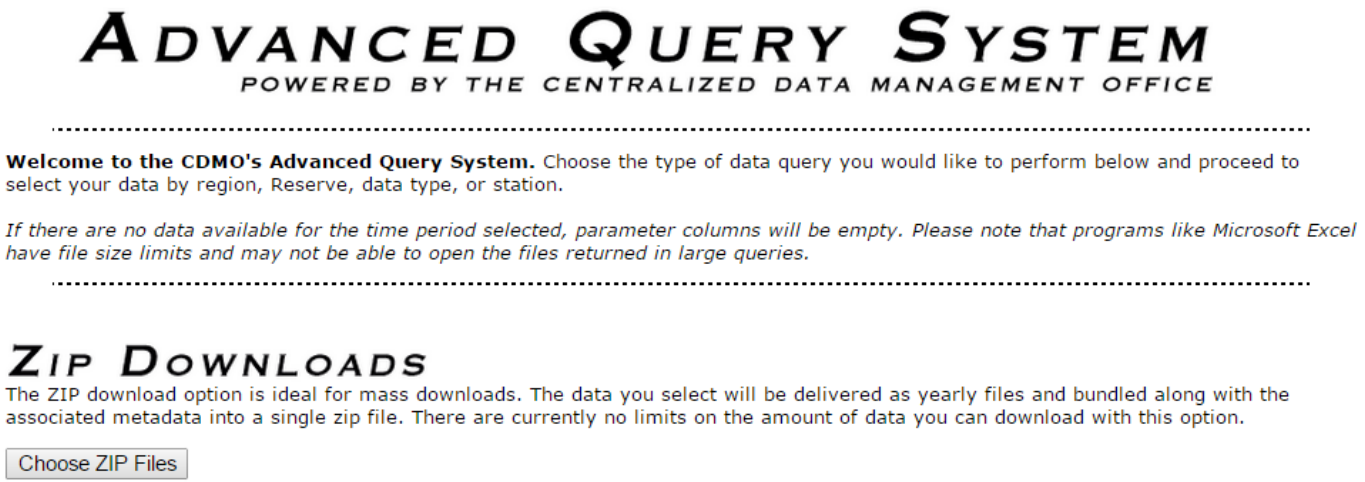
Interactive!

Retrieve SWMP data

We learned how to import SWMP data in the previous session

To review, the easiest approach is to download the data outside of R, then import using the 'import_local' function

Be sure that you use only the zip downloads feature from CDMO - the 'import_local' functions works best with these data



Retrieve SWMP data

Let's import some data for Apalachicola Bay

```
# reload the SWMPr package if you started a new session
library(SWMPr)

# import data
# change this path for the flash drive
path <- 'C:/data/dataset2'
wq_dat <- import_local(path, 'apacpwq')
nut_dat <- import_local(path, 'apacpnut')
met_dat <- import_local(path, 'apaebmet')
```

We've just imported data from 2011–2014 for three stations (apacpwq, apacpnut, apaebmet) and saved them in our workspace

Retrieve SWMP data

But don't take my word for it, take a look at the data!

```
# what are the dimensions of the water quality data?
dim(wq_dat)

## [1] 132035     25

# what are the dimensons of the nutrient data?
dim(nut_dat)

## [1] 48 13

# what are the dimensions of the weather data?
dim(met_dat)

## [1] 133548     23
```

Retrieve SWMP data

View the first six rows

```
# View the first six rows of the met data
head(met_dat) # or tail(met_dat) for last
```

```
##         datetimestamp atemp f_atemp rh f_rh   bp f_bp wspd f_wspd maxwspd
## 1 2011-01-01 00:00:00    15    <0>  94 <0>  1019 <0>     3   <0>        3
## 2 2011-01-01 00:15:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 3 2011-01-01 00:30:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 4 2011-01-01 00:45:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 5 2011-01-01 01:00:00    15    <0>  95 <0>  1018 <0>     3   <0>        4
## 6 2011-01-01 01:15:00    15    <0>  95 <0>  1018 <0>     4   <0>        5
##   f_maxwspd wdir f_wdir sdwdir f_sdwdir totpar  f_totpar totprcp f_totprcp
## 1      <0>   145   <0>       8     <0>     0.8 <1> (CSM)       0      <0>
## 2      <0>   146   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 3      <0>   139   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 4      <0>   140   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 5      <0>   144   <0>       6     <0>     0.8 <1> (CSM)       0      <0>
## 6      <0>   141   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
##   cumprcp f_cumprcp totsorad f_totsorad
## 1       0      <0>        NA      <-1>
## 2       0      <0>        NA      <-1>
## 3       0      <0>        NA      <-1>
## 4       0      <0>        NA      <-1>
## 5       0      <0>        NA      <-1>
## 6       0      <0>        NA      <-1>
```

Retrieve SWMP data

What class is the data?

```
# class of the data
class(met_dat)

## [1] "swmpr"      "data.frame"
```

This tells us that the data are two different classes - 'swmpr' and 'data.frame'

The swmpr object class was developed to work with specific funcitons

```
# what functions/methods work with swmpr objects?
methods(class = 'swmpr')

##  [1] aggregate.swmpr comb.swmpr      decomp.swmpr    hist.swmpr
##  [5] lines.swmpr     na.approx.swmpr plot.swmpr      qaqc.swmpr
##  [9] qaqcchk.swmpr   setstep.swmpr   smoother.swmpr  subset.swmpr
```

Retrieve SWMP data

A useful feature of R is that a class will have both ***data*** and ***attributes***

For the swmpr class, the ***data*** are the raw swmpr data as a data.frame

The ***attributes*** are a list of metadata for the imported data

```
# what attributes are available for a swmpr object
names(attributes(met_dat))

## [1] "names"       "row.names"   "class"       "station"     "parameters"
## [6] "qaqc_cols"   "date_rng"    "timezone"    "stamp_class"

# view the parameters attribute
attr(met_dat, 'parameters')

##  [1] "atemp"    "rh"       "bp"       "wspd"     "maxwspd"  "wdir"
##  [7] "sdwdir"   "totpar"   "totprcp"  "cumprcp"  "totsorad"
```

Organize SWMP data

Now that we have a feel for the data, what needs to be done before we can start analyzing the information?

Perhaps the first organizational tool you want to use is 'qaqc.swmpr':

- Remove observations with a specified QAQC flag value
- Remove QAQC columns: Link to QAQC codes

-5 Outside high sensor range
-4 Outside low sensor range
-3 Data rejected due to QAQC
-2 Missing data
-1 Optional SWMP supported parameter
0 Passed initial QAQC checks
1 Suspect data
2 Open - reserved for later flag
3 Calculated data: non-vented depth/level sensor correction for changes in barometric pressure
4 Historical data: Pre-auto QAQC
5 Corrected data

Retrieve SWMP data

Remember, each parameter has a QAQC column preceded by 'f_'

```
# View the first six rows of the met data
head(met_dat)
```

```
##          datetimestamp atemp f_atemp rh f_rh   bp f_bp wspd f_wspd maxwspd
## 1 2011-01-01 00:00:00    15    <0>  94 <0>  1019 <0>     3   <0>        3
## 2 2011-01-01 00:15:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 3 2011-01-01 00:30:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 4 2011-01-01 00:45:00    15    <0>  95 <0>  1019 <0>     3   <0>        4
## 5 2011-01-01 01:00:00    15    <0>  95 <0>  1018 <0>     3   <0>        4
## 6 2011-01-01 01:15:00    15    <0>  95 <0>  1018 <0>     4   <0>        5
##   f_maxwspd wdir f_wdir sdwdir f_sdwdir totpar  f_totpar totprcp f_totprcp
## 1      <0>   145   <0>       8     <0>     0.8 <1> (CSM)       0      <0>
## 2      <0>   146   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 3      <0>   139   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 4      <0>   140   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
## 5      <0>   144   <0>       6     <0>     0.8 <1> (CSM)       0      <0>
## 6      <0>   141   <0>       7     <0>     0.8 <1> (CSM)       0      <0>
##   cumprcp f_cumprcp totsorad f_totsorad
## 1       0      <0>        NA      <-1>
## 2       0      <0>        NA      <-1>
## 3       0      <0>        NA      <-1>
## 4       0      <0>        NA      <-1>
## 5       0      <0>        NA      <-1>
## 6       0      <0>        NA      <-1>
```

Organize SWMP data

You will have to decide which values to keep

It may be useful to get an idea of the distribution of QAQC flags in the data, use 'qaqcchk'

```
# use qaqcchk to view distribution of qaqc flags
myqaqc <- qaqcchk(met_dat)
```

This function returns a data.frame that summarizes QAQC flags in the data

Organize SWMP data

```
# a subset of results from the qaqcchk function
head(myqaqc)
```

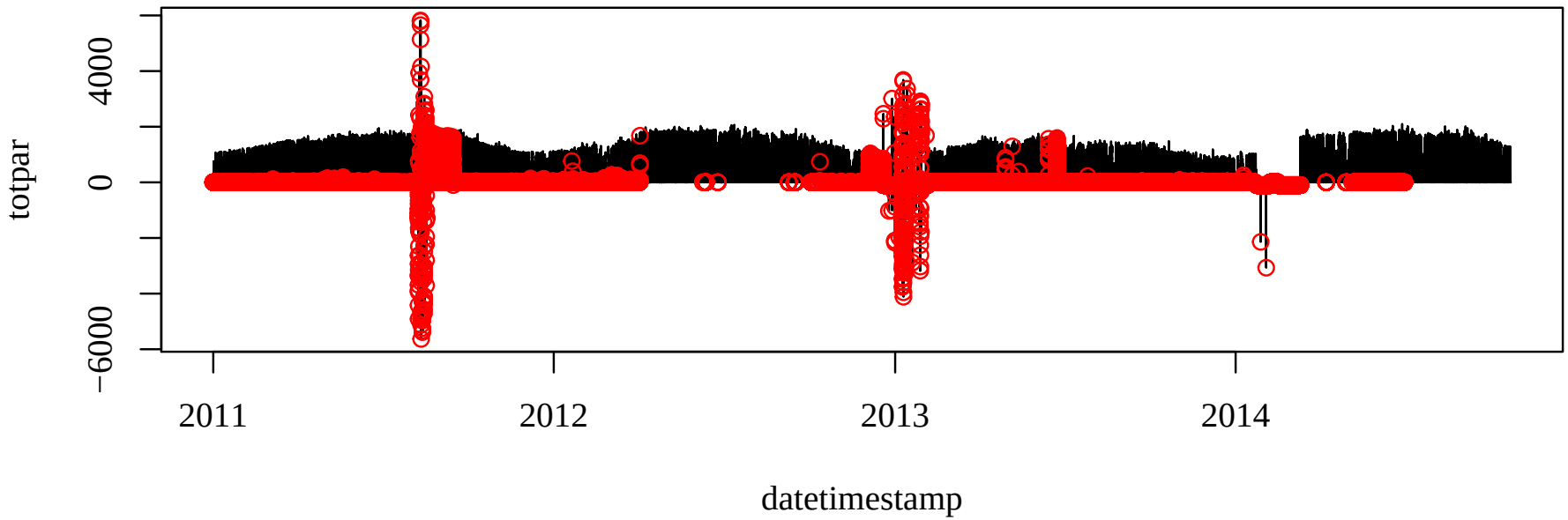
```
##                piece f_atemp f_bp f_cumprcp f_maxwspd f_rh f_sdwdir f_totpar
## 1        <-2> [GPD]       2    2         2         2    2        2        2
## 2        <-3> [GMT]       5   13        70        16    5       16       18
## 3        <-3> [GPD]      15   16        16        16   15       16       16
## 4        <-3> [GPR]      14   14        14        14   14       14       13
## 5        <-3> [SMT]       2   NA       121         3    2        3        2
## 6 <-3> [SQR] (CSM)       3   NA        NA        NA    3       NA     4023
##   f_totprcp f_totsorad f_wdir f_wspd
## 1         2         NA      2      2
## 2        13         NA     16     16
## 3        16         NA     16     16
## 4        14         NA     14     14
## 5        14         NA      3      3
## 6        NA         NA     NA     NA
```

```
# or view all in a separate window
View(myqaqc)
```

Organize SWMP data

A plot of the data may also be useful to view QAQC flags (plot code in script2.R)

Points in red did not pass QAQC



Organize SWMP data

You should have an idea of how you want to handle QAQC values

Next, use the 'qaqc' function...

```
# filter observations by qaqc flags, remove qaqc columns
met_qaqc <- qaqc(met_dat)
```

The default is to keep only observations with a '0' QAQC flag

See the help documentation for the function

```
# view help file
?qaqc
```

Organize SWMP data

View the data after keeping only values that passed QAQC ('0' flag)

```
# data after qaqc processing
head(met_qaqc)
```

```
##          datetimestamp atemp rh   bp wspd maxwspd wdir sdwdir totpar totprcp
## 1 2011-01-01 00:00:00    15 94 1019    3       3  145      8     NA       0
## 2 2011-01-01 00:15:00    15 95 1019    3       4  146      7     NA       0
## 3 2011-01-01 00:30:00    15 95 1019    3       4  139      7     NA       0
## 4 2011-01-01 00:45:00    15 95 1019    3       4  140      7     NA       0
## 5 2011-01-01 01:00:00    15 95 1018    3       4  144      6     NA       0
## 6 2011-01-01 01:15:00    15 95 1018    4       5  141      7     NA       0
##   cumprcp totsorad
## 1       0       NA
## 2       0       NA
## 3       0       NA
## 4       0       NA
## 5       0       NA
## 6       0       NA
```

Organize SWMP data

What if we want to keep all the values, regardless of flag?

```
# keep all values
met_qaqc <- qaqc(met_dat, qaqc_keep = NULL)

head(met_qaqc) # note the totpar column compared to the last example
```

```
##         datetimestamp atemp rh   bp wspd maxwspd wdir sdwdir totpar totprcp
## 1 2011-01-01 00:00:00    15 94 1019    3       3  145      8    0.8       0
## 2 2011-01-01 00:15:00    15 95 1019    3       4  146      7    0.8       0
## 3 2011-01-01 00:30:00    15 95 1019    3       4  139      7    0.8       0
## 4 2011-01-01 00:45:00    15 95 1019    3       4  140      7    0.8       0
## 5 2011-01-01 01:00:00    15 95 1018    3       4  144      6    0.8       0
## 6 2011-01-01 01:15:00    15 95 1018    4       5  141      7    0.8       0
##   cumprcp totsorad
## 1       0       NA
## 2       0       NA
## 3       0       NA
## 4       0       NA
## 5       0       NA
## 6       0       NA
```

Organize SWMP data

If you're not convinced, try removing only the '0' flag

```
# keep only zero flags
to_keep <- c(-5, -4, -3, -2, -1, 1, 2, 3, 4, 5)
met_qaqc <- qaqc(met_dat, qaqc_keep = to_keep)

# does this result make sense??
head(met_qaqc)
```

```
##         datetimestamp atemp rh bp wspd maxwspd wdir sdwdir totpar totprcp
## 1 2011-01-01 00:00:00    NA NA NA   NA      NA   NA     NA    0.8      NA
## 2 2011-01-01 00:15:00    NA NA NA   NA      NA   NA     NA    0.8      NA
## 3 2011-01-01 00:30:00    NA NA NA   NA      NA   NA     NA    0.8      NA
## 4 2011-01-01 00:45:00    NA NA NA   NA      NA   NA     NA    0.8      NA
## 5 2011-01-01 01:00:00    NA NA NA   NA      NA   NA     NA    0.8      NA
## 6 2011-01-01 01:15:00    NA NA NA   NA      NA   NA     NA    0.8      NA
##   cumprcp totsorad
## 1      NA       NA
## 2      NA       NA
## 3      NA       NA
## 4      NA       NA
## 5      NA       NA
## 6      NA       NA
```

Organize SWMP data

We'll continue by using values that passed the QAQC checks

```
# continue with qaqc processed data

# water quality
wq_dat <- qaqc(wq_dat)

# nutrients
nut_dat <- qaqc(nut_dat)

# weather
met_dat <- qaqc(met_dat)
```

Organize SWMP data

What is the next logical step after dealing with QAQC values?

How would we further want to organize the data?

Maybe we want to subset the data...

```
# view help file
?subset.swmpr
```

Organize SWMP data

The subset.swmpr function has several arguments

- swmpr_in: input data (swmpr object)
- subset: dates to keep
- select: parameters to keep
- operator: less than, greater than, etc. if only one date in subset
- rem_rows: remove empty rows
- rem_cols: remove empty columns

Organize SWMP data

The simplest use of 'subset.swmpr' is to remove empty rows and columns

```
# rows, columns in wq_dat
dim(wq_dat)

## [1] 132035     13

# remove empty rows, columns
tmp <- subset(wq_dat, rem_rows = T, rem_cols = T)

# dimensions after removing empty rows, columns
dim(tmp)

## [1] 124273      9
```

About 1000 rows and four columns of missing data! Note that removing rows may create a discontinuous time step...

Organize SWMP data

The 'select' argument of 'subset.swmpr' is used to select parameters of interest - one to many

```
# select the DO column
tmp <- subset(wq_dat, select = 'do_mgl')
head(tmp)
```

```
##          datetimestamp do_mgl
## 1 2011-01-01 00:00:00      6
## 2 2011-01-01 00:15:00      6
## 3 2011-01-01 00:30:00      6
## 4 2011-01-01 00:45:00      6
## 5 2011-01-01 01:00:00      6
## 6 2011-01-01 01:15:00      6
```

Organize SWMP data

The 'select' argument of 'subset.swmpr' is used to select parameters of interest - one to many

```
# select DO and salinity
tmp <- subset(wq_dat, select = c('do_mgl', 'sal'))
head(tmp)
```

```
##          datetimestamp sal do_mgl
## 1 2011-01-01 00:00:00  28      6
## 2 2011-01-01 00:15:00  28      6
## 3 2011-01-01 00:30:00  28      6
## 4 2011-01-01 00:45:00  28      6
## 5 2011-01-01 01:00:00  29      6
## 6 2011-01-01 01:15:00  29      6
```

Note the use of the concatenate function 'c'

Organize SWMP data

The 'subset' argument of 'subset.swmpr' selects a date range

The dates must have a specific format - 'YYYY-mm-dd HH:MM'

```
# select a date range, July 2012
dates <- c('2012-07-01 12:00', '2012-07-31 6:30')
tmp <- subset(wq_dat, subset = dates)
head(tmp) # view first six rows
```

```
##          datetimestamp temp spcond sal do_pct do_mgl depth cdepth level clevel
## 1 2012-07-01 12:00:00   29     34  21     86      6     2     NA    NA     NA
## 2 2012-07-01 12:15:00   28     34  21     83      6     2     NA    NA     NA
## 3 2012-07-01 12:30:00   28     34  21     74      5     2     NA    NA     NA
## 4 2012-07-01 12:45:00   28     34  21     74      5     2     NA    NA     NA
## 5 2012-07-01 13:00:00   28     34  21     73      5     2     NA    NA     NA
## 6 2012-07-01 13:15:00   28     34  22     70      5     2     NA    NA     NA
##   ph turb chlfluor
## 1  8    7       NA
## 2  8    6       NA
## 3  8    5       NA
## 4  8    6       NA
## 5  8    7       NA
## 6  8    8       NA
```

Organize SWMP data

Observations earlier or later than a date can also be selected

This also requires the 'operator' argument – >, <, >=, <= , ==, !=

```
# get observations for 2013
dates <- '2013-01-01 00:00'
tmp <- subset(wq_dat, subset = dates, operator = '>=')
head(tmp)
```

```
##          datetimestamp temp spcond sal do_pct do_mgl depth cdepth level clevel
## 1 2013-01-01 00:00:00   13     42  27     99      9     1     NA    NA     NA
## 2 2013-01-01 00:15:00   13     42  27     99      9     1     NA    NA     NA
## 3 2013-01-01 00:30:00   13     42  27    101      9     1     NA    NA     NA
## 4 2013-01-01 00:45:00   13     42  27    102      9     1     NA    NA     NA
## 5 2013-01-01 01:00:00   13     42  27    100      9     1     NA    NA     NA
## 6 2013-01-01 01:15:00   13     42  27    101      9     1     NA    NA     NA
##   ph turb chlfluor
## 1  8    0       NA
## 2  8    1       NA
## 3  8    1       NA
## 4  8    1       NA
## 5  8    0       NA
## 6  8    0       NA
```

Organize SWMP data

Try a simple application of 'subset' - plot dissolved oxygen and water temperature for October 2014

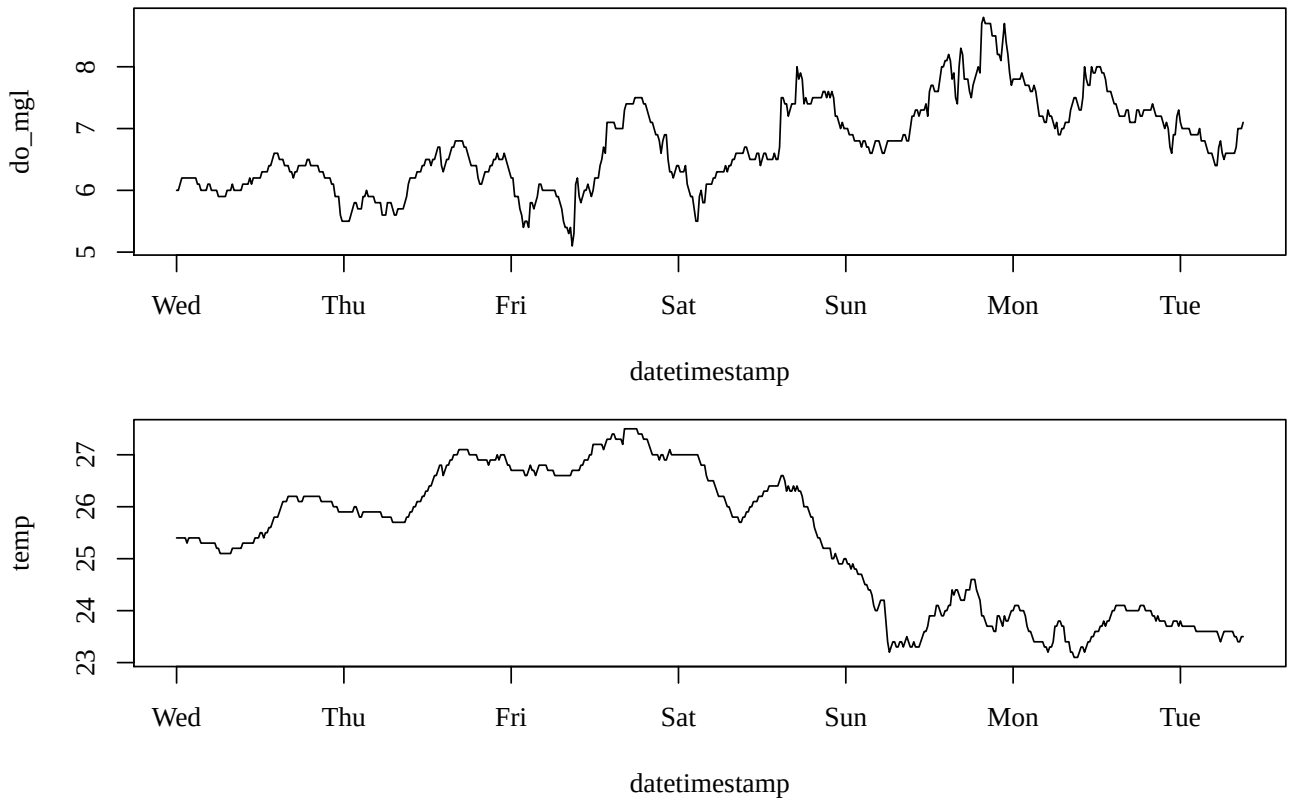
```
# dates and parameters  to select
dates <- '2014-10-01 00:00'
params <- c('do_mgl', 'temp')

# subset
tmp <- subset(wq_dat, select = params, subset = dates, operator = '>=')
head(tmp)
```

```
##         datetimestamp temp do_mgl
## 1 2014-10-01 00:00:00   25      6
## 2 2014-10-01 00:15:00   25      6
## 3 2014-10-01 00:30:00   25      6
## 4 2014-10-01 00:45:00   25      6
## 5 2014-10-01 01:00:00   25      6
## 6 2014-10-01 01:15:00   25      6
```

Organize SWMP data

Try a simple application of 'subset' - plot dissolved oxygen and water temperature for October 2014

```
# plot DO and water temp
plot(do_mgl ~ datetimestamp, data = tmp, type = 'l')
plot(temp ~ datetimestamp, data = tmp, type = 'l')
```



Organize SWMP data

Now that we know how to handle QAQC flags and subset, what else could we do before we analyze?

What if we want to compare time series from different datasets?

Use the 'comb.swmp' and 'setstep.swmp' functions!

```
# help files
?comb.swmpr
?setstep.swmpr
```

Currently, only data from the same reserve can be combined

Organize SWMP data

The ‘setstep.swmpr’ function is used to standardize the time step of a swmpr object

The ‘comb.swmpr’ function is used to combine swmpr objects

‘setstep.swmpr’ is used within ‘comb.swmpr’ so you should not have to use it directly

How is it done??

Organize SWMP data

How can we combine SWMP data?

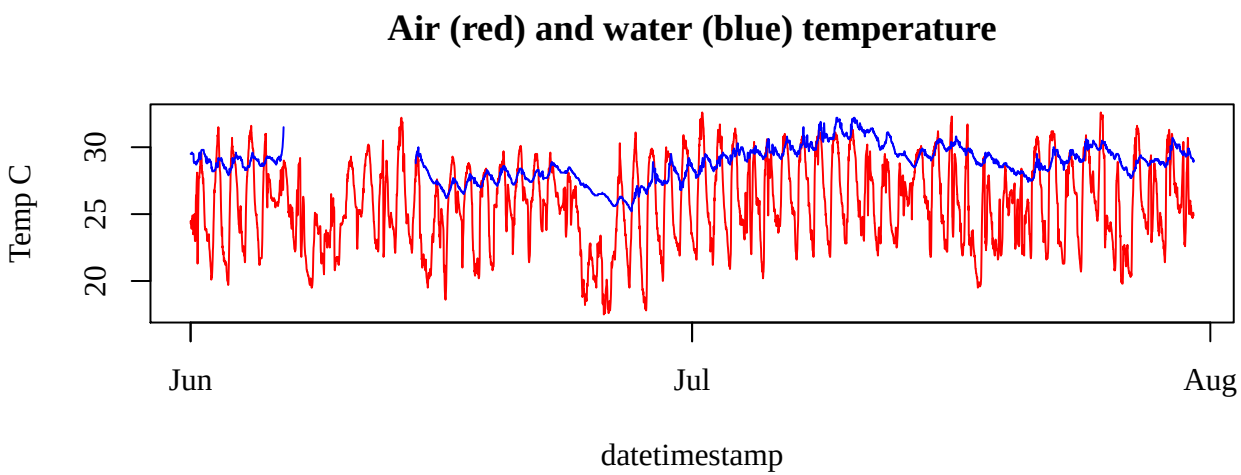
```
# combine water quality and weather data in the same object
tmp <- comb(wq_dat, met_dat)
head(tmp, 3) # first three rows
```

```
##         datetimestamp atemp rh   bp wspd maxwspd wdir sdwdir totpar totprcp
## 1 2011-01-01 00:00:00    15 94 1019    3       3  145      8     NA       0
## 2 2011-01-01 00:15:00    15 95 1019    3       4  146      7     NA       0
## 3 2011-01-01 00:30:00    15 95 1019    3       4  139      7     NA       0
##   cumprcp totsorad temp spcond sal do_pct do_mgl depth cdepth level clevel ph
## 1       0       NA   11     44  28     68      6     2     NA    NA     NA  8
## 2       0       NA   11     44  28     68      6     2     NA    NA     NA  8
## 3       0       NA   11     44  28     68      6     2     NA    NA     NA  8
##   turb chlfluor
## 1    3       NA
## 2    3       NA
## 3    2       NA
```

Organize SWMP data

We now have a swmpr object with data from two stations, why do we want this? Easier plotting...

```
# plot some combined data

# subset date ranges first
dates <- c('2012-06-01 0:0', '2012-07-31 0:0')
to_plot <- subset(tmp, subset = dates)

# plot
plot(atemp ~ datetimestamp, to_plot, type = 'l', col = 'red', ylab = 'Temp C')
lines(to_plot$datetimestamp, to_plot$temp, col = 'blue')
title('Air (red) and water (blue) temperature')
```



Organize SWMP data

Arguments for 'comb.swmpr':

- ... : input swmpr data, separated by comma
- timestep: minutes defining the standardized time step
- differ: maximum difference in minutes for matching observations with original time steps to standardized time steps
- method: how the data are combined using the time stamps - union, intersect, or using a station

Organize SWMP data

Changing the 'timestep' argument can be useful for reducing data volume...

```
# dimension of earlier combined object
dim(tmp)

## [1] 133548     24

# create new object at two hour time step
tmp <- comb(wq_dat, met_dat, timestep = 120)
dim(tmp)

## [1] 16695    24
```

Organize SWMP data

Changing the 'timestep' argument can be useful for reducing data volume...

```
# note the time step in datetimestamp
head(tmp, 4)
```

```
##         datetimestamp atemp rh   bp wspd maxwspd wdir sdwdir totpar totprcp
## 1 2010-12-31 23:00:00    NA NA   NA   NA      NA   NA     NA     NA      NA
## 2 2011-01-01 01:00:00    15 95 1018    3       4  144      6     NA       0
## 3 2011-01-01 03:00:00    15 93 1017    6       7  137      7     NA       0
## 4 2011-01-01 05:00:00    15 95 1017    4       6  146      8     NA       0
##   cumprcp totsorad temp spcond sal do_pct do_mgl depth cdepth level clevel ph
## 1      NA       NA   NA     NA  NA     NA     NA    NA     NA    NA     NA NA
## 2       0       NA   11     44  29     68      6     2     NA    NA     NA  8
## 3       0       NA   11     45  29     66      6     2     NA    NA     NA  8
## 4       0       NA   11     44  28     66      6     1     NA    NA     NA  8
##   turb chlfluor
## 1   NA       NA
## 2    2       NA
## 3    3       NA
## 4    2       NA
```

Organize SWMP data

Caution! Standardizing a time series to a set time step may impose some inaccuracy

Observations are matched as close in time as possible to the standardized time series

By default, 'comb.swmpr' matches the closest observation and discards the rest

Organize SWMP data

'Close' is defined by the 'differ' argument

The 'differ' argument defines the maximum time difference for matching observations to the standardized time step

The maximum allowed value for 'differ' is one half the time step – values beyond this window will create duplicates in your data

Also be careful using small values for differ – data will be lost

Organize SWMP data

Case in point, try standardizing the nutrient time series with 'setstep.swmpr'

```
dim(nut_dat) # initial dimensions

## [1] 48  7

# standardize nutrient time series
tmp <- setstep(nut_dat, timestep = 60, differ = 5)
dim(tmp)

## [1] 25369     7

# remove empty rows, columns
tmp <- subset(tmp, rem_row = T, rem_col = T)
tmp #only four rows!

##         datetimestamp  po4f nh4f no23f chla_n
## 1 2011-02-09 13:00:00 0.004   NA 0.075      8
## 2 2012-07-03 10:00:00 0.004 0.07 0.057      6
## 3 2012-09-05 11:00:00    NA 0.03 0.007      9
## 4 2013-07-09 09:00:00 0.004 0.06 0.036      7
```

Organize SWMP data

A final note about combining data... what about combining data with different time ranges

Consider combining two datasets

- Scenario 1: Time ranges are the same
- Scenario 2: Time ranges are not the same, but there is overlap
- Scenario 3: Time ranges are not the same, there is no overlap

The 'method' argument of 'comb.swmpr' allows flexibility under different scenarios - time range intersect, union, or range of one station

Organize SWMP data

Scenario 1: Time ranges are the same

Don't worry about changing 'method' - it will have no effect

Our previous examples represent scenario 1

Organize SWMP data

Scenario 1: Time ranges are the same

Don't worry about changing 'method' - it will have no effect

Our previous examples represent scenario 1

Scenario 2: Time ranges are not the same, but there is overlap

- union: the default method, use the whole time range of both datasets
- intersect: use only the time range that is shared between the two
- or enter one of the station names, this says use the time range belonging to that station

Organize SWMP data

Scenario 1: Time ranges are the same

Don't worry about changing 'method' - it will have no effect

Our previous examples represent scenario 1

Scenario 2: Time ranges are not the same, but there is overlap

- union: the default method, use the whole time range of both datasets
- intersect: use only the time range that is shared between the two
- or enter one of the station names, this says use the time range belonging to that station

Scenario 3: Time ranges are not the same, there is no overlap

The only method that makes sense is 'union'

Organize SWMP data

Try some examples

```
# first we subset for the example
sub1 <- subset(wq_dat, subset = c('2012-07-01 0:0', '2012-07-31 0:0'))
sub2 <- subset(met_dat, subset = c('2012-07-15 0:0', '2012-08-12 0:0'))

# what are the date ranges?
attr(sub1, 'date_rng')

## [1] "2012-07-01 EST" "2012-07-31 EST"

attr(sub2, 'date_rng')

## [1] "2012-07-15 EST" "2012-08-12 EST"
```

Organize SWMP data

Try some examples

```
# combine using union, this is the default
tmp <- comb(sub1, sub2, method = 'union')

# what is the date range of the combined object?
attr(tmp, 'date_rng')

## [1] "2012-07-01 EST" "2012-08-12 EST"
```

Organize SWMP data

Try some examples

```
# combine using intersect
tmp <- comb(sub1, sub2, method = 'intersect')

# what is the date range of the combined object?
attr(tmp, 'date_rng')

## [1] "2012-07-15 EST" "2012-07-31 EST"
```

Organize SWMP data

Try some examples

```
# combine using the time range in sub1
station <- attr(sub1, 'station')
tmp <- comb(sub1, sub2, method = station)

# what is the date range of the combined object?
attr(tmp, 'date_rng')

## [1] "2012-07-01 EST" "2012-07-31 EST"
```

Organize SWMP data

Now you have an idea of how to organize SWMP data for analysis!

Here's what we did:

- Evaluate and handle QAQC flags in the data
- Subsetting the data to remove empty rows/columns or to select variables or time ranges of interest
- Combining data for comparison or data simplification

Consult the SWMP cookboook for an example workflow!

After lunch... what are some basic ways we can analyze the data?


NERRS / SWMP
Data Analysis Workshop: Time Series
November 17, 2014

Questions??