

# SWMP time series analysis cookbook

Marcus W. Beck, [beck.marcus@epa.gov](mailto:beck.marcus@epa.gov), Todd D. O'Brien, [todd.obrien@noaa.gov](mailto:todd.obrien@noaa.gov)

Nov. 17, 2014

## Contents

|                                                          |          |
|----------------------------------------------------------|----------|
| <b>Introduction</b>                                      | <b>1</b> |
| <b>Basics of R and RStudio</b>                           | <b>2</b> |
| What to do when you start a new session . . . . .        | 2        |
| What to do when you close a session . . . . .            | 2        |
| Installing packages . . . . .                            | 3        |
| Keyboard shortcuts . . . . .                             | 3        |
| Getting help . . . . .                                   | 3        |
| <b>Using the SWMP<sub>r</sub> package</b>                | <b>4</b> |
| A quick workflow for retrieving and organizing . . . . . | 4        |
| Viewing the attributes and subsets of the data . . . . . | 5        |
| Simple numerical summaries . . . . .                     | 6        |
| Aggregating or reducing data volume . . . . .            | 6        |
| Dealing with missing data . . . . .                      | 7        |
| <b>Graphics</b>                                          | <b>7</b> |

## Introduction

This document provides a collection of instructions and scripts that perform specific tasks for working with SWMP data in R. It provides scripts that use functions from the SWMP<sub>r</sub> package, as well as many others available in the base or contributed packages available in R. This cookbook is meant to provide workshop participants with a ‘go-to’ document that builds on content covered in the workshop. Much of the content can be a considered a basic approach to time series and we encourage the exploration of alternative, more specific methods that are available in R. Please see the CRAN task view on [time series](#) for an idea of what is possible. The data used in these scripts have been included with the SWMP<sub>r</sub> package installation or can be downloaded from the workshop website. Please feel free to contact the instructors with questions regarding this cookbook or general questions about time series analysis in R. We encourage any feedback regarding these documents or the content of the workshop.

Workshop website: <http://copepod.org/nerrs-swmp-workshop/>

# Basics of R and RStudio

## What to do when you start a new session

You will need to follow a few simple steps to use data and functions in R if you are starting a new session. Here is a basic workflow of these steps.

- Open a new or saved script that you will use to type code (.r file extension). This is under the File menu on the top for both the basic R install and RStudio.
- Load R packages that you will use. The functions in a package will not be available unless the package is loaded. The package must also be previously installed. You can put the packages you are using at the top of your script.
- Set the working directory. This is where R will load and save files.
- Load any workspace that you were using previously. This is an R specific file format that saves any and all objects that are in a workspace. Ideally, you will have saved a workspace from your previous session.

Here is a sample script of this workflow.

```
# my startup script

# load any packages
library(SWMPPr)
library(ggplot2)

# set the working directory
setwd('C:/my_files')

# load your previous workspace if you saved one
# this must be in the working directory
load(file = 'my_workspace.RData')

# check what you loaded
ls()
```

Data of specific formats can also be loaded in R. Flat text files or comma-separated files are commonly used, although R can import many other types. Use these functions to import data. Don't forget to assign your data to an object.

```
# import data and assign
# data are in the working directory, or use a full path

# import a csv file
dat <- read.csv('my_data.csv', header = T)

# import a text file, separated by commas
dat <- read.table('my_data.txt', header = T, sep = ',')
```

## What to do when you close a session

Make sure you save your work when you close a session! Save the script you're working with using the menu at the top. You will also want to save your data. Just as data of different types can be imported, data can

also be saved in different formats. You will either want to save the whole workspace or individual parts (e.g., a data frame).

```
# save the whole workspace as a .RData file
save(list = ls(), file = 'my_workspace.RData')

# save one object as a .RData file
save(dat, file = 'my_data.RData')

# save as .csv
write.csv(dat, 'my_data.csv')

# save as text file
write.table(dat, 'my_data.txt', sep = ',', row.names = F, quote = F)
```

## Installing packages

R installs and loads packages from its library on your computer. You can see where your library is with `.libPaths()`. Packages that you install from CRAN or elsewhere will go in this library. R will also look here when you load a library. The standard location for downloading packages are on CRAN, although you can also download packages on Github or BioConductor. In the latter case, you will have to first download and load the devtools package off CRAN. Again, you will have to load a package every time you start R if you want to use its functions. You should only have to download a package once, unless you want to install the latest version. Here are some basics to installing and loading a package.

```
# install a package from CRAN
install.packages('ggplot2')

# install packages from Github
install.packages('devtools')
library(devtools)
install_github('fawda123/SWMPPr')
library(SWMPPr)
```

## Keyboard shortcuts

You can view all the keyboard shortcuts in RStudio by clicking on Help, then keyboard shortcuts on the top menu. Also view page 12 on this short intro to R for a list of common shortcuts: [R intro](#).

## Getting help

If all else fails, a Google search will usually point you in the right direction. All of the documentation that comes with R and its packages are available online. A quick search for a function or package will lead you to the documentation. You can also access the help files on your computer in R.

```
# access a help file for a function
help(mean)

# or do this
?mean
```

```
# run the examples in the help file  
example(mean)
```

## Using the SWMPPr package

### A quick workflow for retrieving and organizing

Each of the functions in the SWMPPr package usually have default values for the arguments to increase ease of use. However, be cautious that the default values may not be applicable for your specific dataset. Always consult the help documentation to determine the best options for importing and organizing your data. Also see the [SWMPPr tutorial](#) for more details on each function. The following examples provide generic workflows for importing and organizing data that you have downloaded from CDMO. The `import_local` function is meant to work with data from the [Zip Downloads](#) service from the CDMO. For the purpose of this document, data included with the SWMPPr installation are also shown.

#### Single file

This shows how to import and organize a single file that you downloaded.

```
# import data for apaebmet that you downloaded  
  
# this is an example path with the csv files, change as needed  
path <- 'C:/my_path/'  
  
# import, qaqc, subset to remove empty columns  
dat <- import_local(path, 'apaebmet')  
dat <- subset(qaqc(dat), rem_cols = T)
```

For the next example, the `import_local` function is used to load data included with the SWMPPr distribution. To use it, set the path variable using the `system.file` command shown below (you can see this full path by executing `path` at the command line). Execute both lines to import the data.

```
# import data for apaebmet that comes with SWMPPr  
  
# this is the path for csv example files  
path <- system.file('zip_ex', package = 'SWMPPr')  
  
# import, qaqc, subset to remove empty columns  
dat <- import_local(path, 'apaebmet')  
dat <- subset(qaqc(dat), rem_cols = T)
```

#### Multiple files

This shows how to import and organize multiple files that you downloaded.

```
# import data for multiple stations  
  
# this is an example path with the csv files, change as needed  
path <- 'C:/my_path/'
```

```
# import, combine, qaqc, subset to remove empty columns
wq_dat <- import_local(path, 'apacpwq')
nut_dat <- import_local(path, 'apacpnut')
met_dat <- import_local(path, 'apaebmet')
dat <- comb(wq_dat, nut_dat, met_dat)
dat <- subset(qaqc(dat), rem_cols = T)
```

The remainder of this cookcook will use data from muliple files that were imported and organized in this script.

```
# import data for multiple stations that comes with sumpr

# this is the path for csv example files
path <- system.file('zip_ex', package = 'SWMPPr')

# import, combine, qaqc, subset to remove empty columns
wq_dat <- import_local(path, 'apacpwq')
nut_dat <- import_local(path, 'apacpnut')
met_dat <- import_local(path, 'apaebmet')
dat <- comb(wq_dat, nut_dat, met_dat)
dat <- subset(qaqc(dat), rem_cols = T)
```

## Viewing the attributes and subsets of the data

The imported SWMPPr data have descriptive attributes.

```
# names of the attributes
names(attributes(dat))

# retrieving an attribute
attr(dat, 'date_rng')
```

Viewing the whole dataset is often impractical. Here are some functions for viewing subsets.

```
# View the first 1000 rows
View(dat)

# first six rows, last six rows
head(dat)
tail(dat)

# first n rows, last n rows
head(dat, 30)
tail(dat, 30)

# single variables
dat$do_mgl
dat[, 'do_mgl']

# dimenions
dim(dat)
```

```

nrow(dat)
ncol(dat)

# column 1
dat[, 1]

# row 1
dat[1, ]

# row 1, column 1
dat[1, 1]

```

## Simple numerical summaries

Numerical summaries of the data can be obtained for the entire dataset or single variables. In some cases, you will have to explicitly specify how missing data are handled.

```

# whole dataset
summary(dat)

# individual variables
summary(dat$do_mgl)

# mean, range, var, etc.
# note use of na.rm

mean(dat$do_mgl, na.rm = T)
range(dat$do_mgl, na.rm = T)
var(dat$do_mgl, na.rm = T)
min(dat$do_mgl, na.rm = T)
max(dat$do_mgl, na.rm = T)

# how many missing values?
sum(is.na(dat$do_mgl))

```

## Aggregating or reducing data volume

The SWMP<sub>r</sub> package provides several function for reducing the volume of data. This is useful for not only making the data more manageable, but also providing summary statistics that describe trends.

```

subset

setstep

aggregate

smoother

```

## Dealing with missing data

```
replace with mean  
na.omit  
na.approx
```

## Graphics