SWMPr: An R package for Retrieving, Organizing, and Analyzing Environmental Data for Estuaries

Marcus William Beck[1, *]

1 ORISE Research Participation Program, USEPA National Health and Environmental Effects Research Laboratory, Gulf Ecology Division, 1 Sabine Island Drive, Gulf Breeze, FL 32651, USA

*** beck.marcus@epa.gov**

Abstract

Standardized monitoring programs have vastly improved the quantity and quality of data that form the basis of environmental decision-making. One example in the United States is the System Wide Monitoring Program (SWMP) that was implemented in 1995 by the federally-funded National Estuarine Research Reserve System (NERRS). This program has provided two decades of continuous monitoring data at over 300 fixed stations in 28 estuaries. SWMP data have been used in a variety of applications with the general objective of describing dynamics of estuarine ecosystems to better inform effective coastal management. However, simple tools for processing and evaluating the increasing quantity of data provided by the monitoring network have prevented broad-scale comparisons between systems and, in some cases, simple trend analysis of water quality parameters at individual sites. We describe a new open-source software package, SWMPr, developed in program R for use with SWMP environmental data. The package provides several functions that facilitate data retrieval, organization, and analysis of time series data to describe water quality, weather, and nutrient dynamics in the reserve estuaries. Previously unavailable functions for estuaries are also provided to estimate rates of ecosystem metabolism using the open-water method. Tools included with the SWMPr package have facilitated a cross-reserve comparison of trends, including simple evaluation of changes over time and comparisons of patterns in primary productivity. Overall, the package provides an effective approach to link quantitative information with analysis tools that will greatly inform management programs aimed at coastal protection and restoration.

Introduction

The development of low-cost, automated sensors that have the ability to collect data in near real-time has enabled a proliferation of standardized environmental monitoring programs [1, 2]. These programs provide access to invaluable sources of data that can be used to address a variety of research and management objectives. Applications from automated remote sensors are numerous for aquatic environments with notable examples including prediction of harmful algal blooms and toxicants in freshwater systems [3], development of a hydrometeorological monitoring network to support flash flood warning programs [4], and a national marine buoy network covering large portions of the open ocean and coastal zones of the United States [5]. Automated remote

monitoring programs offer several advantages over traditional site-specific, field-based methods including streamlining of data acquisition, minimizing human error, and reducing the overall cost of the collection process [1]. However, the growing quantity of available information to address relevant questions has contributed to the notion of 'big data' science where analysis limitations are often defined by compututational requirements and signal identification in the presence of noise rather than the availability of information. Growing concerns over the use of adequate quality assurance and control methods, challenges for synthesis and interpretation, and increased emphasis on exploratory-based analytical techniques have characterized applications of data from automated monitoring programs [6,7].

The National Estuarine Research Reserve System (NERRS)[1] is a network of 28 estuarine reserves in the United States that was established by the Coastal Zone Management Act of 1972. The reserves represent different biogeographic regions and estuarine types that were chosen to address multiple goals for long-term research, monitoring, education, and stewardship in support of coastal management. As part of this effort, the System Wide Monitoring Program (SWMP)[2] was implemented in 1995 at over 300 stations at the reserves to provide a robust, long-term monitoring system for water quality, weather, and land-use/habitat change. The SWMP network has provided a continuous source of data collected at near real-time with the intent to evaluate natural and anthropogenic causes of spatiotemporal variation in environmental condition and ecosystem function. These data have been applied in evaluations of relevant characteristics at individual reserves (eg., [8,9]) and differences between reserves (e.g., ecosystem metabolism [10,11], tidal characteristics [12], dissolved oxygen [13]). However, no cross-reserve comparisons have been conducted within the last decade despite the online availability of current SWMP data. NERRS researchers and staff have also expressed a need for quantitative analysis tools to evaluate trends in water quality time series given the quantity and quality of data provided by SWMP [14].

This article describes a software package that was developed to address research needs of the NERRS program using the open-source statistical programming language R [15]. SWMPr (pronounced 'swamper') is an R package that contains functions for retrieving, organizing, and analyzing estuary monitoring data from the System Wide Monitoring Program. Functions provided by SWMPr address many of the common issues working with large datasets created from automated sensor networks, such as data pre-processing to remove unwanted information, combining data from different sources, and exploratory analyses to identify key parameters of interest. Additionally, a cross-reserve comparison of water quality trends and current ecosystem metabolism estimates is provided to illustrate potential applications using the functions in this package. The software is provided specifically for use with NERRS data, although many of the applications are relevant for addressing common challenges working with large datasets.

SWMP overview and data retrieval

Four core data elements are collected through the SWMP monitoring network: abiotic monitoring data, biotic observations, habitat and land use mapping, and sentinel monitoring. The SWMPr package is developed for the continous abiotic monitoring network which includes a majority of the data collected by SWMP. Abiotic elements monitored at each reserve include water quality (water temperature, specific conductivity, salinity, dissolved oxygen concentration, dissolved oxygen saturation, depth, pH, turbidity, chlorophyll fluorescence), weather (air temperature, relative

[1] http://www.nerrs.noaa.gov/
[2] http://nerrs.noaa.gov/RCDefault.aspx?ID=18

humidity, barometric pressure, wind speed, wind direction, photosynthetically active radiation, precipitation), and nutrient data (orthophosphate, ammonium, nitrite, nitrate, nitrite + nitrate, chlorophyll a). Each reserve has no less than four water quality stations and one weather station at fixed locations. Water quality and weather data are collected at 15 minute intervals, whereas nutrient data are collected monthly at each water quality station. All data are made accessible through the Centralized Data Management Office (CDMO) web portal[3], where multiple quality assurance/quality control (QAQC) measures are used to screen the information for accuracy and reliability. The final data include all timestamped observations including relevant QAQC flags with the appropriate qualifier.

The CDMO web portal was established to support priority areas of SWMP that focus on the continuation and advancement of data management. As such, CDMO provides access to over 35 million water quality, weather, and nutrient records that have been authenticated through systematic QAQC procedures. Prior to any data request to the CDMO, the location, parameter type, and date ranges need to be identified based on the analysis needs. All stations in the SWMP network are identified by a 7 or 8 character name that specifies the reserve, station, and parameter type. For example, 'apaebwq' is the water quality identifier ('wq') for the East Bay station ('eb') at the Apalachicola reserve ('apa'). Similarly, a suffix of 'met' or 'nut' would specify the weather (meteorological) or nutrients station. All reserve names, stations, and date ranges for each parameter type can be viewed on the CDMO website. Alternatively, the `site_codes` (all sites) or `site_codes_ind` (single site) functions provided by the SWMPr package can be used to view the same information. As noted below, the computer's IP address must be registered with CDMO before using the data retrieval functions in SWMPr. Web services[4] are provided by CDMO for direct access to SWMP data through http requests, in addition to standard graphical user interface options for selecting data. The data retrieval functions in SWMPr are simple calls to the existing retrieval functions on CDMO web services. For example, the `site_codes` function in SWMPr uses the `exportStationCodesXMLNew` function from the web services to retrieve metadata for all the SWMP sites. The text below describes the data retrieval functions in more detail, including all other functions available in SWMPr.

Structure of the SWMPr package

Installing the package

The SWMPr package was developed for use with the R ($\geq$v3.0.0) statistical programming language [15]. The SWMPr package can be installed from GitHub[5] by executing the following commands at the R console prompt. The package is loaded in the workspace using the `library` command.

```
> install.packages('devtools')
> library(devtools)
> install_github('fawda123/SWMPr')
> library(SWMPr)
```

The SWMPr package was developed by considering a standard workflow that categorizes the functions as one of three steps based on their intended use: retrieving, organizing, and analyzing. Functions for retrieving are used to import the data into R

[3] http://cdmo.baruch.sc.edu/
[4] http://cdmo.baruch.sc.edu/webservices.cfm
[5] https://github.com/fawda123/SWMPr

Table 1. Retrieval functions available from the SWMPr package. Full documentation for each function is in the help file (e.g., execute `?all_params` at the command line).

Function	Description
`all_params`	Retrieve up to 100 records starting with the most recent at a given station, all parameters. Wrapper to `exportAllParamsXMLNew` function on web services.
`all_params_dtrng`	Retrieve records of all parameters within a given date range for a station. Optional argument for a single parameter. Maximum of 1000 records. Wrapper to `exportAllParamsDateRangeXMLNew`.
`import_local`	Import files from a local path. The files must be in a specific format, such as those returned from the CDMO using the zip downloads option.
`import_remote`	Import site data from an independent server. These files have been downloaded from CDMO up to December 2014, processed using SWMPr functions, and uploaded to an Amazon server for quicker import into R.
`single_param`	Retrieve up to 100 records for a single parameter starting with the most recent at a given station. Wrapper to `exportSingleParamXMLNew` function on web services.
`site_codes`	Metadata for all stations, wrapper to `exportStationCodesXMLNew` function on web services.
`site_codes_ind`	Metadata for all stations at a single site, wrapper to `NERRFilterStationCodesXMLNew` function on web services.

as a `swmpr` object class following standard S3 documentation methods [16]. Functions for organizing and analyzing the data provide methods for working with a `swmpr` object. An additional group of 'miscellaneous' functions are included as helpers for the main functions. The following describes a general approach for using each category of functions.

Data retrieval

Three approaches can be used to import SWMP data into R, either through direct download or by importing local data (Table 1). First, functions from the package can be used to import the data directly from the online server using CDMO web services. To do so, the IP address for the computer making the request must be registered with CDMO.[6] The `site_codes` or `site_codes_ind` functions can be used to view the available metadata after a computer is registered with CDMO.

```
> # retrieve metadata for all sites
> site_codes()
>
> # retrieve metadata for a single site
> site_codes_ind('apa')
```

Due to rate limitations on the CDMO server, the retrieval functions return a limited number of records. The functions are more useful for evaluating short time periods,

[6] Follow instructions here: http://cdmo.baruch.sc.edu/webservices.cfm

although these functions could be used iteratively (i.e., with `for` loops) to obtain longer time series. Data retrieval functions to access the CDMO include `all_params`, `all_params_dtrng`, and `single_param`: `all_params` returns the most recent records of all parameters at a station, `all_params_dtrng` returns all records within a date range for all parameters or a single parameter, and `single_param` is identical to `all_params` except that a single parameter is requested.

```
> # all parameters for a station, most recent
> all_params('hudscwq')
>
> # get all parameters within a date range
> all_params_dtrng('hudscwq', c('09/10/2012', '02/8/2013'))
>
> # get single parameter within a date range
> all_params_dtrng('hudscwq', c('09/10/2012', '02/8/2013'),
>   param = 'do_mgl')
>
> # single parameter for a station, most recent
> single_param('hudscwq', 'do_mgl')
```

A second approach for importing SWMP data into R uses the `import_remote` function. This method imports site data from an independent server where SWMP data have been downloaded previously and uploaded as `swmpr` objects in RData binary format. This approach is advantageous because the files can be imported very quickly and an IP address does not need to be registered. The files include almost all available SWMP data for each site up to December 2014. However, the files will not be regularly updated and CDMO should always be used for the most recent and accurate data. The files can be accessed by copying the URL to a web browser with the site name and file extension (e.g., https://s3.amazonaws.com/swmpalldata/acebbnut.RData, see the Supporting Information) or by using the function as follows.

```
> # import a file from an independent server
> import_remote('acebbwq')
```

A third approach for data retrieval is to use the `import_local` function to import data into R that are locally available after downloading from CDMO. This approach is most appropriate for large, specific data requests. The `import_local` function is designed for data from the zip downloads[7] feature in the advanced query section of the CDMO. The function may also work using data from the data export system,[8] although this feature has not been extensively tested. The zip downloads feature is an easy way to obtain data from multiple stations in one request. The downloaded data will be in a compressed folder that includes multiple .csv files by year for a given data type (e.g., apacpwq2002.csv, apacpwq2003.csv, apacpnut2002.csv, etc.). The `import_local` function can be used after the folder is decompressed.

Occasionally, duplicate time stamps are present in the raw data. The `import_local` function handles duplicate entries differently depending on the data type (water quality, weather, or nutrients). For water quality and nutrient data, duplicate time stamps are simply removed. Note that nutrient data often contain replicate samples with similar but not identical time stamps within the span of a few minutes. Replicates with unique time stamps are not removed but can be further processed using `rem_reps`. Weather data prior to 2007 may also contain duplicate time stamps at frequencies for hourly

[7] http://cdmo.baruch.sc.edu/aqs/zips.cfm
[8] http://cdmo.baruch.sc.edu/get/export.cfm

(denoted as '60') and daily ('144') averages, in addition to 15 minute frequencies. Only duplicate values at 15 minutes are averaged for weather data.

```
> # import local data for apaebmet
>
> # this is an example path with the decompressed csv files
> path <- 'C:/my_path/'
>
> # import, do not include file extension
> import_local(path, 'apaebmet')
```

All data retrieval functions return a `swmpr` object that includes relevant data and several attributes describing the dataset. The data include a `datetimestamp` column in the appropriate timezone for a station. Additional columns include parameters for a given data type (weather, nutrients, or water quality) and correspondingg QAQC columns if returned from the initial data request. The attributes for a `swmpr` object include `names` of the dataset, `row.names` of the dataset, `class` (character string indicating `swmpr` and `data.frame`), `station` (7 or 8 characters identifying the station), `parameters` (character vector of data columns, e.g., `'do_mgl'`), `qaqc_cols` (logical `T` or `F` if present or not), `date_rng` (`POSIXct` vector of minimum/maximum dates), `timezone` (text string in country/city format, standard time only), and `stamp_class` (class of `datetimestamp` vector, `POSIXct` or `Date`). Attributes of a `swmpr` object can be viewed as follows.

```
> # import data
> dat <- import_remote('apadbwq')
>
> # view all attributes of dat
> attributes(dat)
>
> # view a single attribute of dat
> attr(dat, 'station')
```

The `swmpr` object class was created for use with specific methods following the S3 object definition approach [16]. A `swmpr` object also secondarily inherits methods from the `data.frame` class, such that common `data.frame` methods also apply to `swmpr` objects. Available methods for the `swmpr` class are described below and can also be viewed:

```
> # view available methods for swmpr class
> methods(class = 'swmpr')
```

A sample dataset can be downloaded for use with the examples below.[9] This dataset has an identical format as the data returned from the zip downloads feature of the CDMO. Processed verions of the raw data are also included with the package as binary data files (RData). These include `swmpr` objects for four stations at Apalachicola Bay: apacpnut, apacpwq, apadbwq, and apaebmet. Information for each file can be viewed in the help documentation (e.g., `?apacpnut`).

Data organizing

The organize functions are used to clean or prepare the imported data for analysis, including viewing and removal of QAQC flags, subsetting, combining replicate nutrient

[9] https://s3.amazonaws.com/swmpexdata/zip_ex.zip

Table 2. Organizing functions available from the SWMPr package. Full documentation for each function is in the help file (e.g., execute `?comb.swmpr` at the command line).

Function	Description
`comb.swmpr`	Combines `swmpr` objects to a common time series using setstep, such as combining the weather, nutrients, and water quality data for a single station. Only different data types can be combined.
`qaqc.swmpr`	Remove QAQC columns and remove data based on QAQC flag values for a `swmpr` object. Only applies if QAQC columns are present.
`qaqcchk.swmpr`	View a summary of the number of observations in a `swmpr` object that are assigned to different QAQC flags used by CDMO. The output can be used to inform further processing.
`rem_reps.swmpr`	Remove replicate nutrient data that occur on the same day. The default is to average replicates.
`setstep.swmpr`	Format data from a `swmpr` object to a continuous time series at a given timestep. The function is used in `comb.swmpr` and can also be used with individual stations.
`subset.swmpr`	Subset by dates and/or columns for a `swmpr` object. This is a method passed to the generic `subset` function provided in the base package.

observations, creating a standardized time series, and combining data of different types (Table 2).

The `qaqc` function is a simple screen to retain observations from the data with specified QAQC flags.[10] Each parameter in the imported `swmpr` object will have a corresponding QAQC column of the same name with the added prefix `f_` (e.g., `do_mgl`, `f_do_mgl`). Values in the QAQC column range from -5 to 5 to indicate the QAQC flag that was assigned by CDMO during initial processing. The QAQC function is used to remove observations in the raw data with given flags, with the default option to retain only values with the `0` QAQC flag (i.e., passed initial CDMO checks). Additionally, simple filters are used to remove obviously bad values, e.g., wind speed values less than zero or pH values greater than 12. Erroneous data entered as -99 are also removed. The function returns the original data with the QAQC columns removed and `NA` (not available) values for observations that do not meet the criteria specified in the function call.

```
> # qaqc screen for a swmpr object, retain only '0'
> qaqc(dat)
>
> # retain all data regardless of flag
> qaqc(dat, qaqc_keep = NULL)
>
> # retain only '0' and '-1' flags
> qaqc(dat, qaqc_keep = c(0, -1))
```

Viewing the number of observations for each parameter that are assigned to a QAQC flag may be useful for deciding how to process the data with `qaqc`. The

[10] CDMO flags: http://cdmo.baruch.sc.edu/data/qaqc.cfm

qaqcchk function can be used to view this information.

```
> # view the number of observations in each QAQC flag
> qaqcchk(dat)
```

Raw nutrient data obtained from the CDMO will usually include replicate samples that were taken within a few minutes of each other. The rem_reps.swmpr function combines nutrient data that occur on the same day to preserve an approximate monthly time step. The datetimestamp column will always be averaged for replicates, but the actual observations will be combined based on the user-supplied function which defaults to the mean. Other suggested functions include the median, min, or max. The entire function call, including treatment of NA, values should be passed to the FUN argument (see the examples). The function is meant to be used after qaqc processing, although it works with a warning if QAQC columns are present.

```
> # get nutrient data
> data(apacpnut)
> dat <- apacpnut
> dat <- qaqc(dat)
>
> # remove replicate nutrient data
> rem_reps(dat)
>
> # use different function to aggregate replicates
> func <- function(x) max(x, na.rm = T)
> rem_reps(dat, FUN = func)
```

A subset method added to the existing generic subset function in R is available for swmpr objects. This function is used to subset the data by date and/or a selected parameter. The date can be a single value or as two dates to select records within the range. The former case requires a binary operator as a character string passed to the argument, such as '>' or '<='. The subset argument for the date(s) must also be a character string of the format YYYY-mm-dd HH:MM for each element (e.g., '2007-01-01 06:30'). Be aware that an error may be returned using this function if the subset argument is in the correct format but the calendar date does not exist, e.g. '2012-11-31 12:00'. Finally, the function can be used to remove rows and columns that do not contain data.

```
> # import data
```

```
> data(apaebmet)
> dat <- apaebmet
>
> # select two parameters from dat
> subset(dat, select = c('rh', 'bp'))
>
> # subset records greater than or equal to a date
> subset(dat, subset = '2013-01-01 0:00', operator = '>=')
>
> # subset records within a date range
> subset(dat, subset = c('2012-07-01 6:00', '2012-08-01 18:15'))
>
> # subset records within a date range, select two parameters
> subset(dat, subset = c('2012-07-01 6:00', '2012-08-01 18:15'),
>   select = c('atemp', 'totsorad'))
>
> # remove rows/columns that do not contain data
> subset(dat, rem_rows = T, rem_cols = T)
```

The `setstep` function formats a `swmpr` object to a continuous time series at a given time step. This function is not necessary for most stations but can be useful for combining data or converting an existing time series to a set interval. The first argument of the function, `timestep`, specifies the desired time step in minutes starting from the nearest hour of the first observation. The second argument, `differ`, specifies the allowable tolerance in minutes for matching existing observations to the defined time steps in cases where the two are dissimilar. Values for `differ` that are greater than one half the value of `timestep` are not allowed to prevent duplication of existing data. Likewise, the default value for `differ` is one half the time step. Time steps that do not match any existing data within the limits of the `differ` argument are not discarded, although a corresponding data value will not be assigned.

```
> # import, qaqc removal
> data(apadbwq)
> dat <- qaqc(apadbwq)
>
> # convert time series to two hour invervals
> # tolerance of +/- 30 minutes for matching existing data
> setstep(dat, timestep = 120, differ = 30)
>
> # convert a nutrient time series to a continuous time series
> # then remove empty rows and columns
> data(apacpnut)
> dat_nut <- apacpnut
> dat_nut <- setstep(dat_nut, timestep = 60)
> subset(dat_nut, rem_rows = T, rem_cols = T)
```

The `comb` function is used to combine multiple `swmpr` objects into a single object with a continuous time series at a given step. The `timestep` function is used internally such that `timestep` and `differ` are accepted arguments for `comb`. All arguments must be called explicitly since an arbitrary number of `swmpr` objects can be used as input. The function combines data by creating a master time series that is used to iteratively merge all `swmpr` objects. The time series for merging depends on the value passed to the

`method` argument. Passing `union` to `method` will create a time series that is continuous from the earliest and latest dates for all input objects. Passing `intersect` to `method` will create a time series that is continuous from the set of dates that are shared between all input objects. Finally, a seven or eight character station name passed to `method` will merge all data based on a continuous time series for the specified station, which must be present in the input data. Currently, combining identical data types from different stations is not possible (e.g., two water quality stations from the same reserve).

```
> # get nuts, wq, and met data as separate objects for the same reserve
> data(apacpnut)
> data(apacpwq)
> data(apaebmet)
> swmp1 <- apacpnut
> swmp2 <- apacpwq
> swmp3 <- apaebmet
>
> # combine nuts and wq data by union
> comb(swmp1, swmp2, method = 'union')
>
> # combine nuts and wq data by intersect
> comb(swmp1, swmp3, method = 'intersect')
>
> # combine nuts, wq, and met data by nuts time series, two hour time step
> comb(swmp1, swmp2, swmp3, timestep = 120, method = 'apacpnut')
```

Data analysis

The analysis functions range from general purpose tools for time series analysis to more specific functions for working with continuous monitoring data in estuaries (Table 3). The general purpose tools are `swmpr` methods that were developed for existing generic functions in the R base installation or relevant packages. These functions include `swmpr` methods for `aggregate`, `filter`, and `approx` to deal with missing or noisy data and more general functions for exploratory data analaysis, such as `plot`, `summary`, and `hist` methods. Decomposition functions, `decomp` and `decomp_cj`, are provided as relatively simple approaches for decomposing time series into additive or multiplicative components. Functions to estimate and plot ecosystem metabolism from combined water quality and weather data are provided by the `ecometab` and `plot_metab` functions. The analysis functions may or may not return a `swmpr` object depending on whether further processing with `swmpr` methods is possible from the output.

The `aggregate` function aggregates parameter data for a `swmpr` object by set periods of observation. This function is most useful for aggregating noisy data to evaluate trends on longer time scales, or to simply reduce the size of a dataset. Data can be aggregated by years, quarters, months, weeks, days, or hours for a user-defined function, which defaults to the mean. A `swmpr` object is returned for the aggregated data, although the `datetimestamp` vector will be converted to a date object if the aggregation period is a day or longer. Days are assigned to the date vector if the aggregation period is a week or longer based on the `round` method for IDate objects data.table package. This approach was used to facilitate plotting using predefined methods for Date and POSIX objects. Additionally, the method of treating NA values for the aggregation function should be noted since this may greatly affect the quantity of data that are returned (see the example below). Finally, the default argument for `na.action` is set to `na.pass` for `swmpr` objects to preserve the time series of the input data.

Table 3. Analysis functions available from the SWMPr package. Full documentation for each function is in the help file (e.g., execute `?aggregate.swmpr` at the command line).

Function	Description
`aggregate.swmpr`	Aggregate `swmpr` objects for different time periods - years, quarters, months, weeks, days, or hours. Aggregation function is user-supplied but defaults to mean.
`aggregate_metab.swmpr`	Aggregate metabolism data from a `swmpr` object. This is primarly used within `plot_metab` but may be useful for simple summaries of raw daily data.
`ecometab.swmpr`	Estimate ecosystem metabolism for a combined water quality and weatehr dataset using the open-water method.
`decomp.swmpr`	Decompose a `swmpr` time series into trend, seasonal, and residual components. This is a simple wrapper to `decompose`. Decomposition of monthly or daily trends is possible.
`decomp_cj.swmpr`	Decompose a `swmpr` time series into grandmean, annual, seasonal, and events components. This is a simple wrapper to `decompTs` in the wq package. Only monthly decomposition is possible.
`hist.swmpr`	Plot a histogram for a `swmpr` object.
`lines.swmpr`	Add lines to an existing `swmpr` plot.
`na.approx.swmpr`	Linearly interpolate missing data (`NA` values) in a `swmpr` object. The maximum gap size that is interpolated is defined as a maximum number of records with missing data.
`plot.swmpr`	Plot a univariate time series for a `swmpr` object. The parameter name must be specified.
`plot_metab.swmpr`	Plot ecosystem metabolism estimates after running `ecometab` on a `swmpr` object.
`plot_summary.swmpr`	Create summary plots of seasonal/annual trends and anomalies for a water quality or weather parameter.
`smoother.swmpr`	Smooth `swmpr` objects with a moving window average. Window size and sides can be specified, passed to `filter`.

```
> # combine, qaqc, remove empty columns
> dat <- comb(swmp1, swmp2, method = 'union')
> dat <- qaqc(dat)
> swmpr_in <- subset(dat, rem_cols = T)
>
> # get mean DO by quarters
> aggregate(swmpr_in, 'quarters', params = c('do_mgl'))
>
> # get mean DO by quarters, remove NA when calculating means
> fun_in <- function(x) mean(x, na.rm = T)
> aggregate(swmpr_in, FUN = fun_in, 'quarters', params = c('do_mgl'))
```

Time series can be smoothed to better characterize a signal independent of noise (Fig. 1). Although there are many approaches to smoothing, a moving window average is intuitive and commonly used. The `smoother` function can be used to smooth parameters in a `swmpr` object using a specified window size. This method is a simple wrapper to `filter`. The `window` argument specifies the number of observations included in the moving average. The `sides` argument specifies how the average is calculated for each observation (see the documentation for `filter`). A value of 1 will filter observations within the window that are previous to the current observation, whereas a value of 2 will filter all observations within the window centered at zero lag from the current observation. As before, the `params` argument specifies which parameters to smooth. See Fig. 1 for the output from the code.

```
> # import data
> data(apadbwq)
> swmp1 <- apadbwq
>
> # qaqc and subset imported data
> dat <- qaqc(swmp1)
> dat <- subset(dat, subset = c('2012-07-09 00:00', '2012-07-24 00:00'))
>
> # filter
> test <- smoother(dat, window = 50, params = 'do_mgl')
>
> # plot to see the difference
> plot(do_mgl ~ datetimestamp, data = dat, type = 'l')
> lines(test, select = 'do_mgl', col = 'red', lwd = 2)
```

A common issue with any statistical analysis is the treatment of missing values. Missing data can be excluded from the analysis, included but treated as true zeroes, or interpolated based on similar values. In either case, an analyst should have a strong rationale for the chosen method. A common approach used to handle missing data in time series analysis is linear interpolation. A simple curve fitting method is used to create a continuous set of records between observations separated by missing data. A challenge with linear interpolation is an appropriate gap size for fitting missing observations. The ability of the interpolated data to approximate actual trends is a function of the gap size. Interpolation between larger gaps are less likely to resemble patterns of an actual parameter, whereas interpolation between smaller gaps are more likely to resemble actual patterns. An appropriate gap size limit depends on the unique characteristics of specific datasets or parameters. The `na.approx` function can be used to interpolate gaps in a `swmpr` object. A required argument for the function is `maxgap`

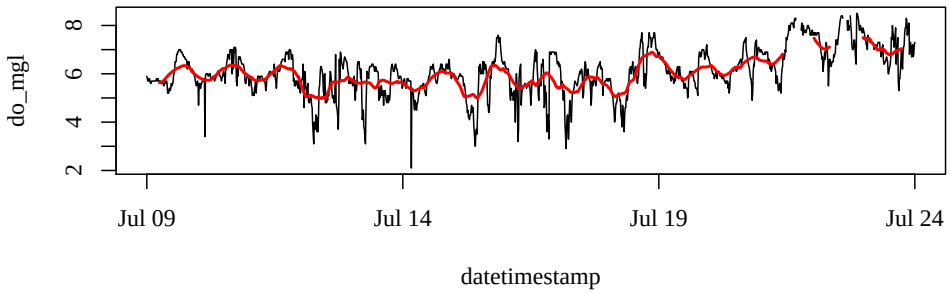

Fig. 1. Raw and smoothed dissolved oxygen data for a two-week period after using the `smoother` function.

which defines the maximum gap size for interpolation. See Fig. 2 for the output from the following code.

```
> # get data
> data(apadbwq)
> swmp1 <- apadbwq
>
> # qaqc and subset imported data
> dat <- qaqc(swmp1)
> dat <- subset(dat, subset = c('2013-01-22 00:00', '2013-01-26 00:00'))
>
> # interpolate, maxgap of 10 records
> test <- na.approx(dat, params = 'do_mgl', maxgap = 10)
>
> # interpolate maxgap of 30 records
> test2 <- na.approx(dat, params = 'do_mgl', maxgap = 30)
>
> # plot for comparison
> par(mfrow = c(3, 1))
> plot(do_mgl ~ datetimestamp, dat, main = 'Raw', type = 'l')
> plot(do_mgl ~ datetimestamp, test, col = 'red',
+   main = 'Interpolation - maximum gap of 10 records', type = 'l')
> lines(dat, select = 'do_mgl')
> plot(do_mgl ~ datetimestamp, test2, col = 'red',
+   main = 'Interpolation - maximum gap of 30 records', type = 'l')
> lines(dat, select = 'do_mgl')
```

The `decomp` function is a simple wrapper to `decompose` that separates a time series into additive or multiplicative components describing a trend, cyclical variation (e.g., daily or seasonal), and the remainder. The additive decomposition assumes that the cyclical component of the time series is stationary (i.e., the variance is constant), whereas a multiplicative decomposition accounts for non-stationarity. By default, a moving average with a symmetric window is used to filter the seasonal component. Alternatively, a vector of filter coefficients in reverse time order can be supplied (see the help documentation for `decompose`).

The `decompose` function requires a `ts` object with a specified frequency as input.

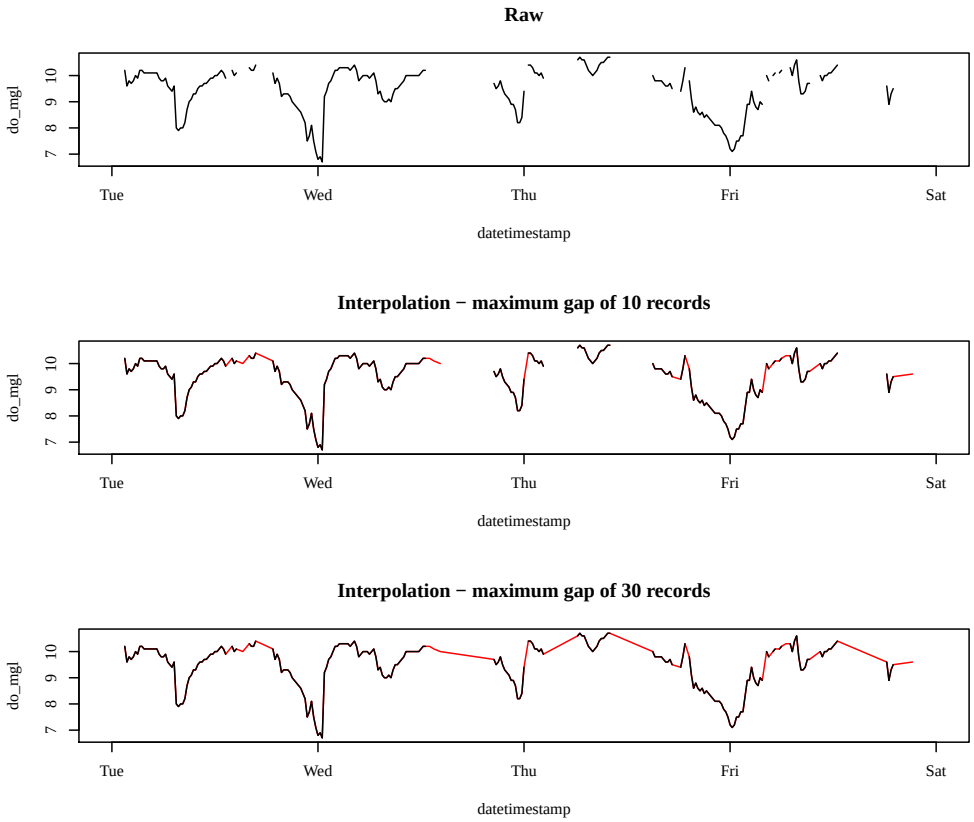

Fig. 2. Examples illustrating use of the `na.approx` function to fill gaps of different sizes in a dissolved oxygen time series for a four day period.

The `decomp` function converts the input `swmpr` vector to a `ts` object prior to `decompose`. This requires an explicit input defining the frequency of the parameter in the time series. For example, the frequency of a parameter with diurnal periodicity would be 96 if the time step is 15 minutes (4 * 24). The frequency of a parameter with seasonal periodicity would be 35040 (4 * 24 * 365). For simplicity, character strings of `'daily'` or `'seasonal'` can be supplied in place of numeric values. A starting value of the time series must be supplied in the latter case. Use of the `setstep` function is also required to standardize the time step prior to decomposition. Note that the `decompose` function is a relatively simple approach and alternative methods should be investigated if a more sophisticated decomposition is desired. Fig. 3 is an example of the `decomp` function.

```
> # get data
> data(apadbwq)
> swmp1 <- apadbwq
>
> # subset for daily decomposition
> dat <- subset(swmp1, subset = c('2013-07-01 00:00', '2013-07-31 00:00'))
>
> # decomposition and plot
> test <- decomp(dat, param = 'do_mgl', frequency = 'daily')
> plot(test)
```

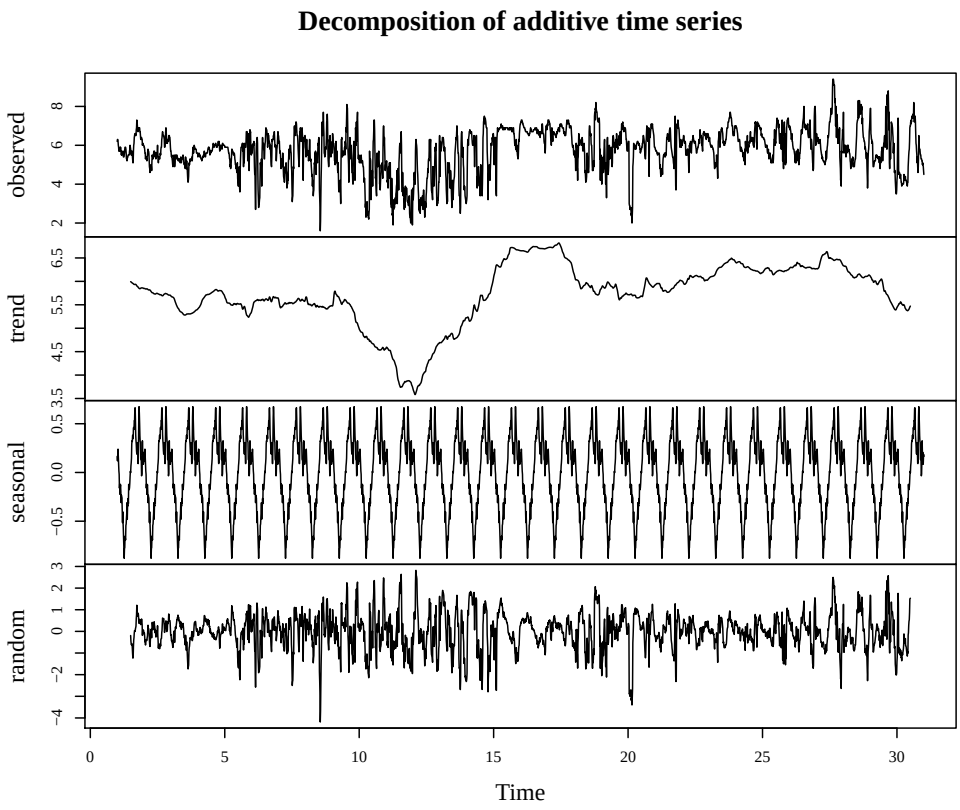

Fig. 3. An additive decomposition of dissolved oxygen into a trend, seasonal, and random component using the `decomp` function.

An alternative approach to time series decomposition is provided by the `decomp_cj` function, which is a simple wrapper to the `decompTs` function in the wq package. Theory describing this method is provided by Cloern and Jassby [17]. The function is similar to `decomp.swmpr` with a few key differences. The `decomp.swmpr` function decomposes the time series into a trend, seasonal, and random component, whereas the current function decomposes into the grandmean, annual, seasonal, and events components. For both functions, the random or events components, respectively, can be considered anomalies that do not follow the trends in the remaining categories. The `decomp_cj` function provides only a monthly decomposition, which is appropriate for characterizing relatively long-term trends. This approach works best for nutrient data that are typically obtained on a monthly cycle. The function will also work with continuous water quality or weather data but note that the data must first aggregated on the monthly scale before decomposition. Additional arguments passed to `decompTs` can be used with `decomp_cj`, such as `startyr`, `endyr`, and `type`. Values passed to `type` are `mult` (default) or `add`, referring to multiplicative or additive decomposition. Fig. 4 shows the results from the `decomp_cj` function applied to a multi-year chlorophyll time series.

```
> # get data
```

```
> data(apacpnut)
> dat <- apacpnut
> dat <- qaqc(dat, qaqc_keep = NULL)
>
> # decomposition of chl, ggplot
> decomp_cj(dat, param = 'chla_n')
```

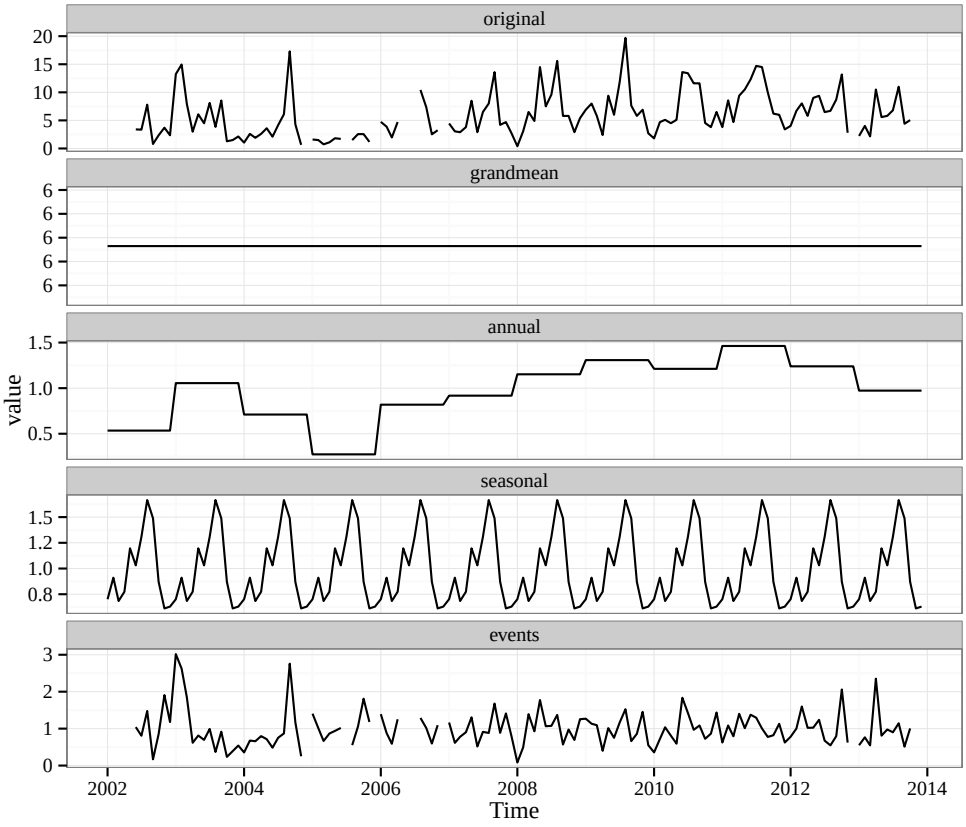

Fig. 4. Additive decomposition of a multi-year chlorophyll time series into the grandmean, annual, seasonal, and events components using the `decomp_cj` function.

Several graphics showing seasonal and annual trends for a given SWMP parameter can be obtained using the `plot_summary` function. The plots include monthly distributions, monthly anomalies, and annual anomalies in multiple formats. Anomalies are defined as the difference between the monthly or annual average from the grand mean for the parameter. Monthly anomalies are in relation to the grand mean for the same month across all years. All data are aggregated for quicker plotting. Nutrient data are based on monthly averages, whereas weather and water quality data are based on daily averages. Cumulative precipitation data are based on the daily maximum. The function returns a graphics object (Grob) of multiple ggplot objects. An interactive Shiny application [18] that uses this function is available (see the Supporting Information).

```
> ## import data
> data(apacpnut)
> dat <- qaqc(apacpnut)
>
> ## plot
> plot_summary(dat, param = 'chla_n', years = c(2007, 2013))
```

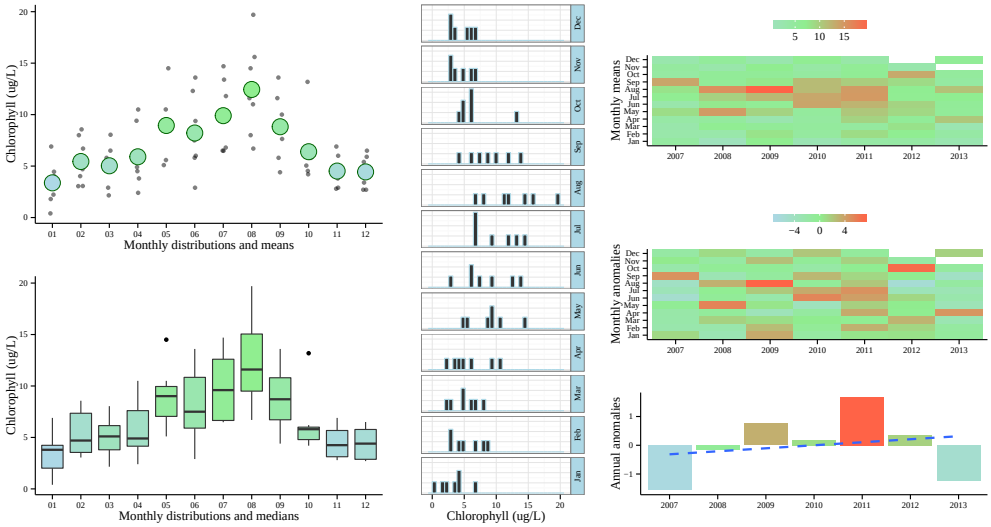

Fig. 5. Summaries of a multi-year chlorophyll time series using the `plot_summary` function. Summaries include monthly distributions (means on top left, quantiles on bottom left), monthly histograms (center), monthly means by year (top right), deviation from monthly means (middle right), and annual trends as deviations from the grand mean (bottom right)

Estimates of ecosystem metabolism provide a measure of overall system productivity to evaluate whether an ecosystem is a net source or sink of organic material. The open-water method [19] is a common approach to quantify metabolism using a mass balance equation that describes the change in dissolved oxygen over time from the balance between photosynthetic and respiration processes, corrected using an empirically constrained air-sea gas diffusion model (see Ro and Hunt [20], Thebault et al. [21]). The diffusion-corrected dissolved oxygen (DO) flux estimates are averaged separately over each day and night of the time series. The nighttime average DO flux is used to estimate respiration rates, while the daytime DO flux is used to estimate net primary production. To generate daily integrated rates, respiration rates are assumed constant such that hourly night time DO flux rates are multiplied by 24. Similarly, the daytime DO flux rates are multiplied by the number of daylight hours, which varies with location and time of year, to yield net daytime primary production. Respiration rates are subtracted from daily net production estimates to yield gross production rates. The metabolic day is considered the 24 hour period between sunsets on two adjacent calendar days

The `ecometab` function is used to implement the open-water method with a combined water quality and weather dataset [22]. Several assumptions must be met for a valid interpretation of the results. In general, the dissolved oxygen time series is assumed to represent the same water mass over time. Tidal advection may have a significant influence on the time series, which can contribute to a significant amount of

noise in metabolic estimates. The extent to which tidal advection influences the dissolved oxygen signal depends on various site-level characteristics and an intimate knowledge of the site may be required. Areal rates for gross production and total respiration are based on volumetric rates normalized to the depth of the water column at the sampling location, which is assumed to be well-mixed, such that the water quality sensor is reflecting the integrated processes in the entire water column (including the benthos). Water column depth is calculated as the mean value of the depth variable across the time series in the `swmpr` object. Depth values are floored at one meter for very shallow stations and 0.5 meters is also added to reflect the practice of placing sensors slightly off of the bottom. Additionally, the air-sea gas exchange model is calibrated with wind data either collected at, or adjusted to, wind speed at 10 m above the surface. The metadata should be consulted for exact height. Other assumptions may apply and relevant resources should be consulted to ensure appropriate application of the open-water method (see [23, 24]).

The following is an example that shows use of the function from a combined water quality and weather data set. The results can be plotted using `plot_metab` (Fig. 6).

```
> ## import water quality and weather data
> data(apadbwq)
> data(apaebmet)
>
> ## qaqc, combine
> wq <- qaqc(apadbwq)
> met <- qaqc(apaebmet)
> dat <- comb(wq, met)
>
> ## estimate metabolism
> res <- ecometab(dat, trace = FALSE)
> plot_metab(res)
```

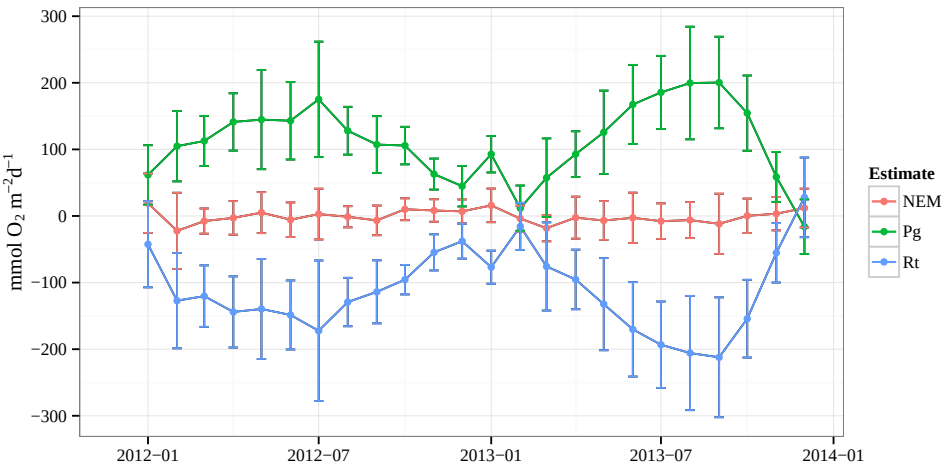

Fig. 6. Monthly aggregations of ecosystem metabolism estimates (net ecosystem metabolism, gross production, and total respiration) for combined water quality and weather data at Apalachicola Bay, Florida.

Table 4. Miscellaneous functions available from the SWMPr package. Most are used within the main functions above but may be useful for more customized evaluationg of SWMP data. Full documentation for each function is in the help file (e.g., execute `?calckl` at the command line).

Function	Description
`calckl`	Estimate the reaeration coefficient for air-sea gas exchange. This is only used within the `ecometab` function.
`map_reserve`	Create a map of all stations in a reserve using the ggmap package.
`metab_day`	Identify the metabolic day for each approximate 24 period in an hourly time series. This is only used within the `ecometab` function.
`param_names`	Returns column names as a list for the parameter type(s) (nutrients, weather, or water quality). Includes QAQC columns with `f_` prefix. Used internally in other functions.
`parser`	Parses html returned from CDMO web services, used internally in retrieval functions.
`swmpr`	Creates object of `swmpr` class, used internally in retrieval functions.
`time_vec`	Converts time vectors to `POSIXct` objects with correct time zone for a site/station, used internally in retrieval functions.

Miscellaneous functions

Several additional functions are provided that do not fit the above categories (Table 4). These functions are generally used within the main functions but may be useful for more customized evaluation of SWMP data.

For brevity, only the `reserve_map` function is discussed. This function can be used to create a map with all stations at a reserve by passing arguments to functions in the ggmap package [25]. The current function is limited to Google maps of four types that can be set with the `map_type` argument: terrain (default), satellite, roadmap, or hybrid. The `zoom` argument may have to be chosen through trial and error depending on the spatial extent of the reserve. See the help documentation for the `ggmap` function for more info on `zoom`.

```
> # plot the stations at Jacques Cousteau reserve
> map_reserve('jac')
```

Applications using the SWMPr package

The ability to evaluate environmental characteristics between estuaries within the NERRS program has been greatly improved using functions in the SWMPr package. This section describes three examples using the SWMPr package to illustrate the improved ability to synthesize and evaluate multi-year time series of estuarine data.

First, the open-water method for estimating metabolism was applied to nearly all co-located water quality and weather sites at the NERRS reserve for all years of available data. The results are provided primarily to illustrate ease of use of the functions and secondarily to provide an update to results described in Caffrey [10] and Caffrey [11] using the most recent SWMP data. Additionally, previous methods to

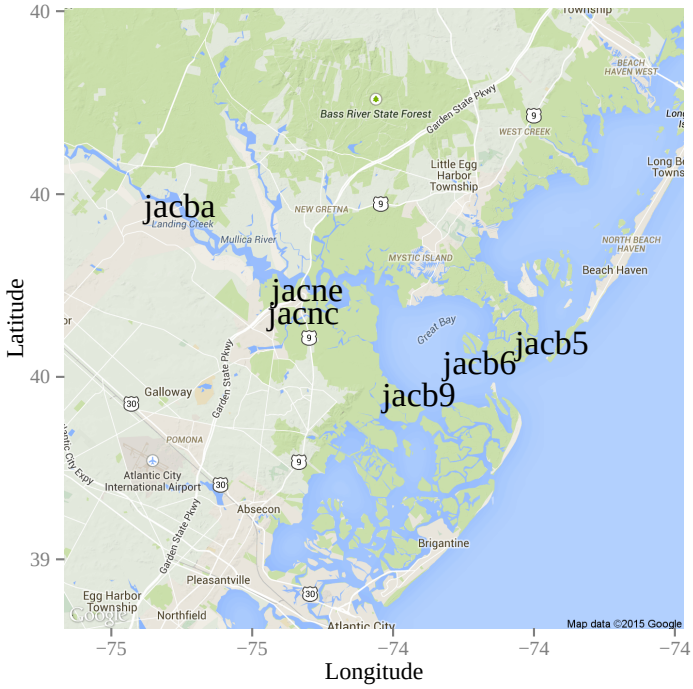

Fig. 7. Locations of all sites at the Jacques Cousteau reserve using the `map_reserve` function.

evaluate metabolism at NERRS reserves did not incorporate *in situ* weather data to estimate the air-sea gas exchange model. Caffrey [10] and Caffrey [11] applied the open-water method to estimate ecosystem metabolism using five years of water quality observations at two sites at each of the NERRS reserves. Additionally, the air-sea gas exchange model assumed a constant value for the reaeratio coefficientt. This coefficient provides an estimate of the rate of air-sea gas exchange that varies as a function of wind speed, temperature, barometric pressure, salinity, and depth of the water column. The inclusion of weather data in the calculation allows for a more precise estimate of air-sea gas exchange and consequently more reliable estimates of ecosystem metabolism (see Caffrey et al. [22] for details).

Water quality and weather observations from January 1995 to December 2014 for all NERRS sites were obtained through a bulk data request using the zip downloads feature of CDMO. All csv files for each station were imported into R using the `import_local` function, processed using the `setstep` and `qaqc` functions, then saved on a local hard drive as binary RData files (see here for the R script). This resulted in a single `swmpr` object for each parameter at each site. All files were then uploaded to a remote server. An additional R script was executed that retrieved and processed combined the water quality and weather data for each site to esimate metabolism. For brevity, only two sites at each reserve were evaluated. Mean annual values at each site, organized by region, are shown in Fig. 8, whereas decadal comparisons are shown in Table 5. All sites were generally net heterotrophic across the range of observations (i.e., sink of organic matter, in agreement with Caffrey [10]), although differences were observed in early (i.e., 1995-2004) as compared to recent (2005-2014) time periods. Overall, the results indicate that between-region and within-site differences in metabolism are apparent and varying by time period, such that a more comprehensive evaluation of factors that influence metabolic rates is needed. More importantly, the use

of the data retreival, synthesis, and analysis functions to create the results shown in Fig. 8 and Table 5 illustrates the utility provided by the SWMPr package.

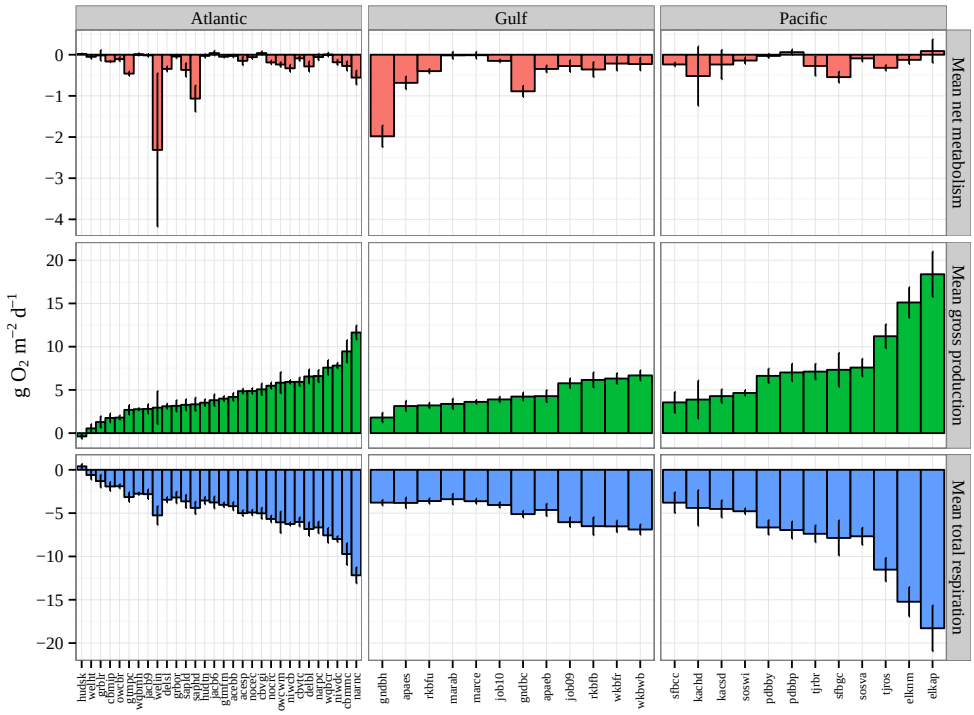

Fig. 8. Aggregated estimates of net metabolism, gross production, and total respiration for two sites at each NERRS reserve. Values are daily integrated estimates as mean annual values averaged across all years with 95% confidence intervals. Two sites were chosen from each reserve that had the longest available time series. Regions are approximations of geographic coordinates and are not exact, e.g., Old Woman Creek (Lake Erie) is Atlantic.

The second and third examples are two Shiny web applications [26] created using the SWMPr package that illustrate summaries and comparisons of SWMP data within and between reserves (see the Supporting Information). The second application evaluates trends in SWMP data within and between sites using an interactive map. Trends between reserves can be viewed using the map, whereas trends at individual sites can be viewed by clicking on a map location. Site-levels trends are described below the map using a simple linear regression to show an increase or decrease in values over time. Trends on the map at each station are plotted as circles that identify the direction and significance of the trend. The trend direction is blue for decreasing and red for increasing. The significance is indicated by radius of the circle and color shading where larger points with darker colors indicate a strong trend. The third application provides graphical summaries of water quality, weather, or nutrient station data at individual stations. The drop down menus can be used to select the station, date range, and parameter for plotting. Plots are based on daily averages for each parameter. The data used for each application are similar to those used to estimate ecosystem metabolism described above.

Table 5. Trends in metabolism for two sites at each of the NERRS reserves. Values are averages of mean annual estimates for each period of observation (1994-2004 and 2005-2014). Bold values indicate an increase from the first period, whereas italic values indicate a decrease. Regions are approximations of geographic coordinates and are not exact, e.g., Old Woman Creek (Lake Erie) is Atlantic.

Site	NEM[a]		Pg		Rt	
	1995-2004	2005-2014	1995-2004	2005-2014	1995-2004	2005-2014
Atlantic						
acebb	-0.04	**-0.03**	5.11	*3.81*	-5.15	**-3.82**
acesp	-0.26	**-0.1**	4.68	**4.93**	-4.94	*-5.03*
cbmip	-0.14	*-0.17*	1.02	**1.91**	-1.16	*-2.08*
cbmmc	-0.43	**-0.24**	6.06	**10.13**	-6.48	*-10.38*
cbvgi	0.13	*0*	6.56	*4.47*	-6.43	**-4.46**
cbvtc	-0.15	**-0.07**	5.58	**6.07**	-5.73	*-6.14*
delbl	-0.59	**-0.16**	6.68	*6.51*	-7.27	**-6.67**
delsl	-0.37	**-0.33**	2.84	**3.22**	-3.21	*-3.55*
grblr		-0.02		1.28		-1.31
grbor		-0.04		3.16		-3.2
gtmfm	-0.09	**-0.04**	3.36	**4.19**	-3.45	*-4.23*
gtmpc	-0.48	**-0.45**	1.95	**2.91**	-2.43	*-3.37*
hudsk	0.02	*0.02*	-0.36	*-0.39*	0.38	**0.41**
hudtn	-0.03	*-0.03*	3.75	*3.43*	-3.78	**-3.46**
jacb6	0.05	*0.03*	3.41	**3.93**	-3.36	*-3.89*
jacb9	0.01	*-0.02*	1.8	**3.1**	-1.79	*-3.12*
narnc	-0.64	**-0.53**	12.64	*11.33*	-13.25	**-11.86**
narpc	-0.14	**-0.03**	7.36	*6.3*	-7.5	**-6.32**
niwcb	-0.46	**-0.28**	5.5	**6.1**	-5.95	*-6.38*
niwdc	-0.13	*-0.2*	7.28	**8.02**	-7.41	*-8.23*
nocec	-0.05	*-0.06*	4.86	**4.88**	-4.92	*-4.94*
nocrc	-0.19	*-0.19*	5.93	*5.31*	-6.12	**-5.5**
owcbr	-0.17	**-0.09**	1.45	**1.9**	-1.62	*-1.99*
owcwm	-0.36	**-0.19**	7.02	*5.36*	-7.38	**-5.54**
saphd	-1.28	**-1**	1.89	**3.77**	-3.17	*-4.77*
sapld	-0.16	*-0.43*	2.32	**3.56**	-2.49	*-3.99*
welht	-0.07	**-0.04**	0.36	**0.63**	-0.43	*-0.67*
welin	-1.87	*-2.49*	3.61	*2.68*	-5.48	**-5.18**
wqbcr	-0.01	**0.01**	8.13	*7.41*	-8.14	**-7.4**
wqbmh	0.02	*0*	2.82	*2.76*	-2.79	**-2.75**
Gulf						
apaeb	-0.35	*-0.35*	4.19	**4.31**	-4.54	*-4.67*
apaes	-0.71	**-0.67**	3.35	*3.05*	-4.06	**-3.73**
gndbc	-1.02	**-0.88**	3.5	**4.3**	-4.51	*-5.18*
gndbh	-1.81	*-2*	2.19	*1.77*	-4	**-3.77**
job09	-0.34	**-0.25**	5.32	**5.95**	-5.66	*-6.2*
job10	-0.15	*-0.15*	3.58	**4.03**	-3.72	*-4.2*
marab		-0.02		3.38		-3.4
marce		-0.01		3.62		-3.63
rkbfb	-0.28	*-0.37*	5.28	**6.25**	-5.56	*-6.62*
rkbfu	-0.33	*-0.41*	2.95	**3.24**	-3.28	*-3.64*
wkbfr	-0.29	**-0.14**	6.27	**6.35**	-6.57	**-6.49**
wkbwb	-0.29	**-0.16**	6.81	*6.54*	-7.1	**-6.69**
Pacific						
elkap	0.03	**0.11**	14.7	**19.84**	-14.67	*-19.74*
elknm	-0.24	**-0.08**	11.95	**16.37**	-12.18	*-16.45*
kachd	0.2	*-0.66*	7.31	*3.2*	-7.12	**-3.86**
kacsd	-0.06	*-0.26*	5.15	*4.2*	-5.21	**-4.45**
pdbbp	0.01	**0.08**	6.29	**7.31**	-6.27	*-7.23*
pdbby	-0.07	**-0.01**	8.39	*5.92*	-8.47	**-5.93**
sfbcc		-0.24		3.56		-3.79
sfbgc		-0.55		7.32		-7.87
sosva	-0.2	**-0.05**	6.98	**7.83**	-7.18	*-7.87*
soswi	-0.28	**-0.08**	4.41	**4.74**	-4.69	*-4.82*
tjrbr	-1.33	**-0.17**	7.72	*7.06*	-9.05	**-7.22**
tjros	-0.26	*-0.34*	10.18	**11.61**	-10.44	*-11.96*

[a]NEM: net ecosystem metabolism, Pg: gross production, Rt: total respiration, all values in g O_2 m^{-2} d^{-1} as annual averages.

Summary

The ability of management and research programs to address critical environmental issues is highly dependent on the quality of data used to inform decision making. Standardized monitoring programs have vastly improved the ability to evaluate factors that influence a range of conditions, leading to more comprehensive assessment of site-specific characteristics and more informed decisions to manage environmental resources. The System Wide Monitoring Program has provided twenty years of continuous monitoring of environmental characteristics at over over 300 stations within the 28 estuaries of the National Estuarine Research Reserve System. This monitoring network establishes a foundation for more effective coastal management by providing standardized data to address spatiotemporal variation in natural and anthropogenic characteristics that influence environmental condition. Although the data provided by SWMP are unique among coastal observing systems and have been used in a variety of applications [8–10, 12, 13], the capacity of NERRS researchers and staff to more effectively evaluate SWMP data could be greatly improved using the SWMPr package to address common challenges working with large datasets.

The SWMPr package provides several functions to retrieve, organize, and analyze SWMP data to more effectively evaluate environmental trends. The package is designed to augment, rather than replace, existing data retrieval programs by providing a bridge betwen the raw data and the analysis software through its numerous data retrieval functions (Table 1). Established QAQC methods and data processing techniques are also enhanced with SWMP by functions that filter observations for different QAQC flags (`qaqc`) and subset by selected dates or variables (`subset`). Additionally, cumbersome challenges comparing differents datasets are addressed by the `setstep` and `comb` functions that standardize time steps and combine the data, respectively. Finally, the analysis functions provide numerous tools to implement common analyses for time series and more specific methods for water quality data. In particular, the `ecometab` function can be used to estimate daily integrated rates of ecosystem metabolism using the open-water method [19, 22]. The above analysis (see Applications using the SWMPr package) provided a cursory update of metabolism estimates for each the NERRS estuaries using recent data to evaluate trends over time. Although further evaluation of the data are needed, particularly regarding assumptions of the open-water method and tidal effects, the results could be used in a more comprehensive evaluation of factors that influence estuary metabolism. Further development of the SWMPr package will consider modifying existing and including additional functions to more effectively integrate data analysis with the quality of information provided by SWMP and NERRS.

Supporting Information

Trends in SWMP parameters

Evaluate trends within and between reserves using an interactive map:
https://beckmw.shinyapps.io/swmp_comp

Monthly and annual summary of SWMP parameters

Summarize trends in SWMP parameters for an individual site:
https://beckmw.shinyapps.io/swmp_summary/

SWMP data as R binary files

Quickly access all SWMP data from 1995 through 2014. See the help file for `import_remote` for more information: https://s3.amazonaws.com/swmpalldata/

Acknowledgments

I acknowledge the significant efforts of NERRS researchers and staff for providing access to high-quality monitoring data. Thanks particularly to Dwayne Porter and Melissa Ide from CDMO for maintaining the online database. Thanks to Marie Bundy and Nikki Dix for providing me the opportunity to share this package with the broader NERRS community. Thanks to Todd O'Brien for the inspiration for the online widgets in the supporting information. Thanks to Mike Murrell and Jim Hagy III for assistance with documentation and implementation of the ecosystem metabolism functions. The views expressed in this article are those of the authors and do not necessarily reflect the views or policies of the U.S. Environmental Protection Agency. The use of trade names or products does not constitute endorsement by the US Government.

References

1. Glasgow HB, Burkholder JM, Reed RE, Lewitus AJ, Kleinman JE. Real-time remote monitoring of water quality: a review of current applications, and advancements in sensor, telemetry, and computing technologies. Journal of Experimental Marine Biology and Ecology. 2004;300(1-2):409–448.

2. Fries DP, Ivanov SZ, Bhanushali PH, wilson JA, Broadbent HA, Sanderson AC. Broadband, low-cost, coastal sensor nets. Oceanography. 2008;20(4):150–155.

3. Reed RE, Burkholder JM, Allen EH. Current online monitoring technology for surveillance of algal blooms, potential toxicity, and physicalechemical structure in rivers, reservoirs, and lakes. In: American Water Works Association Manual M57, Algae. Denver, Colorado: American Water Works Association; 2010. p. 1–24.

4. National Weather Service, National Oceanic and Atmospheric Administration. Hydrometeorological Automated Data System website; 2015. http://www.nws.noaa.gov/oh/hads/. (Accessed March, 2015).

5. NDBC (National Data Buoy Center). National Oceanic and Atmospheric Administration's National Data Buoy Center; 2015. http://www.ndbc.noaa.gov/. (Accessed March, 2015).

6. Campbell JL, Rustad LE, Porter JH, Taylor JR, Dereszynski EW, Shanley JB, et al. Quantity is nothing without quality: Automated QA/QC for streaming environmental sensor data. BioScience. 2013;63(7):574–585.

7. Millie DF, Weckman GR, Young WA, Ivey JE, Fries DP, Ardjmand E, et al. Coastal 'big data' and nature-inspired computation: prediction potentials, uncertainties, and knowledge derivation of neural netowrks for an algal metric. Estuarine, Coastal and Shelf Science. 2013;125:57–67.

8. Bulthius DA. Distribution of seagrasses in a north Puget Sound estuary - Padilla Bay, Washington, USA. Aquatic Botany. 1995;50(1):99–105.

9. Dix NG, Phlips EJ, Gleeson RA. Water quality changes in the Guana Tolomato Matanzas National Estuarine Research Reserve, Florida, associated with four tropical storms. Journal of Coastal Research. 2008;55(SI):26–37.

10. Caffrey JM. Production, respiration and net ecosystem metabolism in U.S. estuaries. Environmental Monitoring and Assessment. 2003;81(1-3):207–219.

11. Caffrey JM. Factors controlling net ecosystem metabolism in U.S. estuaries. Estuaries. 2004;27(1):90–101.

12. Sanger DM, Arendt MD, Chen Y, Wenner EL, Holland AF, Edwards D, et al. A synthesis of water quality data: National Estuarine Research Reserve System-wide Monitoring Program (1995-2000). Charleston, South Carolina: National Estuarine Research Reserve Technical Report Series 2002:3. South Carolina Department of Natural Resources, Marine Resources Division Contribution No. 500; 2002.

13. Wenner E, Sanger D, Arendt M, Holland AF, Chen Y. Variability in dissolved oxygen and other water-quality variables within the National Estuarine Research Reserve System. Journal of Coastal Research. 2004;45(SI):17–38.

14. System-Wide Monitoring Program Data Analysis Training. SWMP Data Analysis Training Workshop provided at the 2014 NERRS/NERRA Annual Meeting, November 17, 2014; 2014. http://copepod.org/nerrs-swmp-workshop/.

15. RDCT (R Development Core Team). R: A language and environment for statistical computing, v3.1.2. R Foundation for Statistical Computing, Vienna, Austria; 2014. http://www.R-project.org.

16. Wickham H. Advanced R. Boca Raton, Florida: Chapman and Hall, CRC Press; 2014.

17. Cloern JE, Jassby AD. Patterns and scales of phytoplankton variability in estuarine-coastal ecosystems. Estuaries and Coasts. 2010;33(2):230–241.

18. Chang W, Cheng J, Allaire J, Xie Y, McPherson J. shiny: Web Application Framework for R; 2015. R package version 0.11.1. Available from: http://CRAN.R-project.org/package=shiny.

19. Odum HT. Primary production in flowing waters. Limnology and Oceanography. 1956;1(2):102–117.

20. Ro KS, Hunt PG. A new unified equation for wind-driven surficial oxygen transfer into stationary water bodies. Transactions of the American Society of Agricultural and Biological Engineers. 2006;49(5):1615–1622.

21. Thébault J, Schraga TS, Cloern JE, Dunlavey EG. Primary production and carrying capacity of former salt ponds after reconnection to San Francisco Bay. Wetlands. 2008;28(3):841–851.

22. Caffrey JM, Murrell MC, Amacker KS, Harper J, Phipps S, Woodrey M. Seasonal and inter-annual patterns in primary production, respiration and net ecosystem metabolism in 3 estuaries in the northeast Gulf of Mexico. Estuaries and Coasts. 2013;37(1):222–241.

23. Kemp WM, Testa JM. Metabolic balance between ecosystem production and consumption. In: Wolanski E, McLusky DS, editors. Treatise on Estuarine and Coastal Science. New York: Academic Press; 2012. p. 83–118.

24. Needoba JA, Peterson TD, Johnson KS. Method for the quantification of aquatic primary production and net ecosystem metabolism using in situ dissolved oxygen sensors. In: Tiquia-Arashiro SM, editor. Molecular Biological Technologies for Ocean Sensing. New York: Springer; 2012. p. 73–101.

25. Kahle D, Wickham H. ggmap: A package for spatial visualization with Google Maps and OpenStreetMap; 2013. R package version 2.3. Available from: http://CRAN.R-project.org/package=ggmap.

26. Chang W, Cheng J, Allaire J, Xie Y, McPherson J. shiny: Web Application Framework for R; 2015. R package version 0.11.1. Available from: http://CRAN.R-project.org/package=shiny.