




An R package for correcting continuous water quality monitoring data for drift

Andrew R. Shaughnessy  ·
Christopher G. Prener ·
Elizabeth A. Hasenmueller

Received: 19 September 2018 / Accepted: 4 June 2019
© Springer Nature Switzerland AG 2019

Abstract Continuous water quality monitoring instruments are used to understand the chemical and physical behaviors of aquatic environments over time. However, the data generated from these instruments are susceptible to inaccuracies due to drift that can occur between site visits. While there are several software packages available to correct drift in water quality data, these packages are often proprietary, expensive, and/or do not offer the user control over the data corrections. This paper describes *driftR*, an R package that corrects drift in water quality data. *driftR* implements either one- or two-point variable data corrections based on the number of standards used to calibrate the sensor of interest, then linearly interpolates the correction over the period of interest. This program gives control to users to correct each

parameter in a way that is ideal for their unique studies and offers a free, reproducible method for drift correction.

Keywords Water quality · R · Continuous monitoring data · Drift corrections

Introduction

In situ water quality monitoring devices are ideal for assessing aquatic environments because these instruments offer chemical and physical data sets at high-resolution (sampling intervals to < 1 min). These devices have been used in a variety of studies to investigate a wide range of aquatic systems including freshwater habitats, marine environments, and wastewater influents and effluents, to name a few. For example, flood flow components for rivers have been determined using continuous specific conductivity data (Pellerin et al. 2008; Bhaskar and Welty 2015; Hasenmueller et al. 2017). Continuous monitoring of chloride has been used to study the short- and long-term impacts of road de-icing chemicals on groundwater quality (Robinson and Hasenmueller 2017). Roberts et al. (2007) investigated primary production and respiration in headwater streams using continuous dissolved oxygen (DO) and temperature measurements. In the Chesapeake Bay, the relationship between nutrient fluxes and algal blooms was studied using in situ monitoring of chlorophyll a,

A. R. Shaughnessy (✉)
Saint Louis University, St. Louis, MO, USA
e-mail: ars637@psu.edu

C. G. Prener
Department of Sociology and Anthropology, Saint Louis University, St. Louis, MO, USA
e-mail: chris.prener@slu.edu

E. A. Hasenmueller
Department of Earth and Atmospheric Sciences, Saint Louis University, St. Louis, MO, USA
e-mail: elizabeth.hasenmueller@slu.edu

A. R. Shaughnessy
Present Address: Department of Geosciences, Pennsylvania State University, 503 Deike Building, University Park, State College, PA 16802, USA

nitrate, ammonium, and phosphate (Glibert et al. 2008). Jimenez-Montealegre et al. (2002) utilized DO, temperature, and pH to calibrate a dynamic model investigating nitrogen transformations and fluxes in a fish pond. There are many other examples of studies that have relied on in situ water quality data.

Currently, a suite of sensors is available to measure water quality parameters temperature, conductivity, turbidity, DO, pH, selected ions, and chlorophyll *a*, among others. These monitoring devices are marketed by numerous companies including Xylem Analytics (YSI loggers), In-Situ Inc., HORIBA Ltd., and Onset Computer Corporation (HOBO loggers). The devices are frequently used by the private sector, academia, and government agencies (e.g., the United States Geological Survey (USGS) and National Oceanic and Atmospheric Administration (NOAA)), which sometimes make their continuous water quality data publicly available for use. An issue with continuous water quality monitoring devices is that the sensors are susceptible to inaccuracies due to calibration drift and fouling, because instruments are often deployed for weeks at a time between calibrations. Instrument drift is any incremental error in the measured value of a given parameter compared to the true value of that parameter. Biofouling occurs when organic films build up on a sensor's surface and cause interference with measurements. Delauney et al. (2010) illustrated the effects of biofouling on drift in fluorescence measurements.

Many government agencies, academics, and private sector groups have independently attempted to develop methods to correct instrument drift. For example, the USGS corrects their continuous water quality monitoring data sets for calibration drift and fouling using the internally developed program Automated Data Processing System (ADAPS). This program linearly interpolates the drift correction (i.e., the difference between the actual value of a calibration standard and the instrument's reading of that calibration standard after the end of the monitoring deployment) over the entire data set (Wagner et al. 2006). While the ADAPS program effectively adjusts water quality data for calibration drift and fouling, this program is not publicly available. There are also proprietary software packages that are available to the public for drift correction, but these programs are expensive (i.e., hundreds to thousands of dollars) and often do not offer the user control over the correction applied. Moreover,

the full equations implemented in the calibration drift corrections are not made accessible by either the USGS or proprietary software developers, making it impossible for the public to reproduce and/or implement their own corrections based on standard methods. Open-source platforms for developing scientific tools, data platforms, and method documentation are growing in popularity and availability (Buck 2015; Lowndes et al. 2017). Well-documented workflows increase the ease of collaboration and reproducibility. For example, the Ocean Health Index utilizes R to code data preparation and model development as well as Git and GitHub for version control and collaboration (Lowndes et al. 2017).

There is currently a suite of packages available in R that pertain to the access and analysis of water quality data (e.g., `waterData` (Ryberg and Vecchia 2012), `dataRetrieval` (Hirsch and De Cicco 2015), `loadflex` (Appling et al. 2015), `EGRET` (Hirsch et al. 2010), `SWMPR` (Beck 2016)); however, there are currently no R packages to correct water quality data gathered by continuous in situ monitoring devices for drift. The `ODM TOOLS` Python package (Horsburgh et al. 2015) does address water quality data processing, but only allows for one-point drift corrections. The USGS recommends utilizing a two-point drift correction when the dynamic range of a parameter is large (Wagner et al. 2006). Thus, this paper describes an R package, `driftR`, that offers the user control over the drift correction and implements free and reproducible methods for correcting data.

Design

Sensor drift and fouling can substantially impact the quality of a data set (Fig. 1). Through analyzing drift data from 100 deployments of YSI 6600 V2 Multi-Parameter Water Quality Sondes, we found that sensors drift as much as 206% over deployments of approximately 2 weeks (see Table 1; data from Robinson and Hasenmueller (2017) and unpublished data sets). The minimum average sensor drift ($\sim 1\%$) was observed for pH electrode sensors, while highest average drift ($\sim 30\%$) was observed for ion selective electrode (ISE) sensors like chloride (Table 1).

We based `driftR` on the drift corrections used by the USGS (Wagner et al. 2006) and updated by

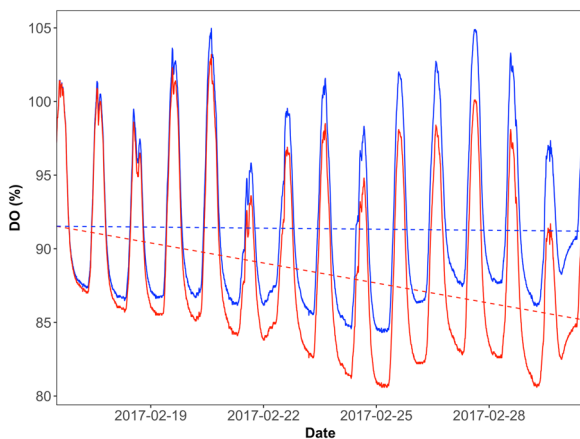


Fig. 1 Continuous (i.e., 5-min data intervals) DO data plotted for a 2-week monitoring period. Uncorrected DO data (red) were corrected (blue) using a one-point drift correction in `driftR` (i.e., the `dr_correctOne()` function). The dashed lines illustrate ~ 6% drift in the data by the end of the monitoring period based on a linear regression of the data

Hasenmueller (2011). In detail, each `driftR` correction is based on linear drift over time:

$$f_t = \left(\frac{t}{\sum t} \right) \tag{1}$$

where f_t is the correction factor, t is the time interval that has passed since the instrument was originally deployed, and $\sum t$ is the total deployment time. This equation calculates time-weighted correction factors that are used to adjust for increasing drift over the monitoring period. In other words, data taken closer to

the initial calibration are corrected less than data taken toward the end of the monitoring period. These correction factors are then used in subsequent calculations for one- or two-point calibrations.

For one-point calibrations, the data correction is expressed as:

$$C = m + f_t \cdot (s_i - s_f) \tag{2}$$

Here, C is the drift-corrected water quality parameter value, m is the uncorrected value, s_i is the value of the calibration standard, and s_f is the value read by the instrument for the calibration standard after the total deployment time (i.e., $\sum t$). One-point calibrations are typically used for parameters such as specific conductivity or DO (see Fig. 1), which either do not drift significantly between calibrations or for which it is difficult to create multiple standards for field use (Wagner et al. 2006).

For two-point calibrations, intermediate calibration standard correction factors for the low (a_t) and high (b_t) standards must first be calculated:

$$a_t = a_i + f_t \cdot (a_i - a_f) \tag{3a}$$

$$b_t = b_i - f_t \cdot (b_i - b_f) \tag{3b}$$

where a_i and b_i are the values of the low and high calibration standards, respectively, and a_f and b_f are the values read by the instrument for the low and high calibration standards, respectively, after the total deployment time (i.e., $\sum t$). These intermediate

Table 1 Calibration drift for various water quality parameters measured by YSI 6600 V2 Multi-Parameter Water Quality Sonde instruments over 8 to 21 days of deployment

Parameter	Std. value ^a	Calibration drift			n^b
		Max (%)	Mean (%)	Std. dev. (%)	
One-point calibrations					
DO	99%	13.37	2.38	2.16	92
Specific conductivity	1.000 mS/cm	97.00	7.26	18.82	88
Two-point calibrations ^c					
pH, low standard	7.00	7.14	1.05	1.48	100
pH, high standard	10.00	9.70	1.11	1.85	98
Chloride, low standard	10.00 mg/L	206.20	30.67	34.19	90
Chloride, high standard	1000.00 mg/L	141.60	26.71	27.40	85

^aThe values of the calibration standards used for the data presented in the table

^bThe number of deployments analyzed for each parameter

^cDrift statistics calculated separately for the high and low calibration standards

Table 2 Functions provided in `driftR` with descriptions and examples

Function	Description and example(s)
<code>dr_read()</code>	Import and format water quality data <pre>dr_read(file = waterData.csv, instrument = ``Sonde``, defineVar = TRUE, cleanVar = TRUE, case = ``snake``)</pre>
<code>dr_factor()</code>	Create correction factors <pre>dr_correct(df, corrFactor = corFac, dateVar = Date, timeVar = Time, keepDateTime = TRUE)</pre>
<code>dr_correctOne()</code>	One-point drift correction <pre>dr_correctOne(df, sourceVar = DO, cleanVar = DO_corr, calVal = 94, calStd = 99, factorVar = corFac)</pre>
<code>dr_correctTwo()</code>	Two-point drift correction <pre>dr_correctTwo(df, sourceVar = pH, cleanVar = pH_corr, calValLow = 7.01, calStdLow = 7, calValHigh = 11.8, calStdHigh = 10, factorVar = corFac)</pre>
<code>dr_drop()</code>	Drop observations from the data set <pre>dr_drop(df, head = 6, tail = 9) dr_drop(df, dateVar = Date, timeVar = Time, from = ``1/12/18``, to = ``1/16/18``) dr_drop(df, exp = turbidity < 0)</pre>
<code>dr_replace()</code>	Replace individual parameters for specified data with NA <pre>dr_replace(df, sourceVar = pH, overwrite = TRUE, dateVar = Date, timeVar = Time, from = ``1/12/18``, to = ``1/16/18``) dr_replace(df, sourceVar = pH, cleanVar = pH_replaced, overwrite = FALSE, exp = pH <= 2)</pre>

values are then passed into the two-point correction formula to determine the drift-corrected water quality parameter value:

$$C = \left(\frac{m - a_t}{b_t - a_t} \right) \cdot (b_i - a_i) + a_i \quad (3c)$$

Two-point calibrations are commonly used for water quality parameters that experience more drift, such as ISE sensors for chloride, nitrate, and ammonium. Two-point calibrations are also important for sensors that experience a large dynamic range of concentrations in the field (e.g., chloride sensors deployed in streams during road de-icing applications or DO sensors deployed in lakes during algal blooms) and/or when it is easy to create multiple standards for field use (Wagner et al. 2006).

A three-point calibration is also possible, though this type of drift correction is more complicated. Three-point data corrections are used when drift is expected to be non-linear; however, most water quality sensors are designed by manufacturers to respond linearly (Wagner et al. 2006). Thus, this type of calibration was not incorporated into the `driftR` package.

Using `driftR`

The `driftR` package is available for download from CRAN using the base function `install.packages()`. Once `driftR` has been installed, the base function `library()` can be used to call `driftR`'s functions. Alternatively, the development version of `driftR` is available via GitHub. What follows is a brief overview of `driftR`'s five core functions, as described in Table 2.

Importing and formatting water quality data

To use `driftR`, water quality data must be imported into R in a standard format. The `dr_read()` function is designed to simultaneously import and correctly format data files generated by water quality monitoring instruments including YSI 6600 V2 Multi-Parameter Water Quality Sondes, YSI EXO2 Multi-Parameter Water Quality Sondes, and HOBO U24 Fresh Water Conductivity Data Loggers. Specifying "Sonde", "EXO", or "HOBO" in the `instrument` argument of the function denotes

Fig. 2 A sample data object, printed in R, with modified output from a YSI 6600 V2 Multi-Parameter Sonde. These data can be previewed from within `driftR` by accessing the `sondeRaw` object

```
# A tibble: 1,527 x 11
  Date      Time  Temp SpCond  pH  pHmV Chloride Turbidity  DO
  <chr>    <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 9/18/2015 12:10:49 14.76 0.754 7.18 -36.4 51.22 3.7 92.65
2 9/18/2015 12:15:50 14.64 0.750 7.14 -34.1 49.62 -0.2 93.73
3 9/18/2015 12:20:51 14.57 0.750 7.14 -33.9 49.75 -0.1 93.95
4 9/18/2015 12:25:51 14.51 0.749 7.13 -33.9 50.32 -0.2 93.23
5 9/18/2015 12:30:51 14.50 0.749 7.13 -33.6 50.74 0.0 92.74
6 9/18/2015 12:35:51 14.63 0.749 7.13 -33.5 50.84 0.0 93.71
7 9/18/2015 12:40:51 14.69 0.749 7.13 -33.6 50.66 -0.2 94.56
8 9/18/2015 12:45:51 14.66 0.749 7.12 -33.3 50.23 -0.2 94.16
9 9/18/2015 12:50:52 14.65 0.749 7.12 -33.3 50.49 -0.2 93.58
10 9/18/2015 12:55:51 14.69 0.749 7.12 -33.1 50.04 -0.2 93.80
# ... with 1,517 more rows
```

that the data were collected via each of the previously mentioned instruments, respectively. This `instrument` argument provides special handling tailored to those instruments, each of which exports data with formatting that requires modification before use with statistical software. Additionally, importing data via `dr_read()` can optionally “tidy” variable names by removing spaces and special characters, making it easier to interact with the variables in R.

However, `driftR` can still be used regardless of the data source as long as those data follow the standard formatting guidelines outlined here:

1. The date and time variables are stored as character data,
2. There is only one header row (i.e., no metadata are stored outside of the primary header), and
3. All of the data to be corrected are stored as numeric data.

Source data that do not fit the formatting guidelines outlined above need to be reformatted using functions from either base R or the more modern `tidyverse` family of packages (Wickham 2017), until they resemble the example shown in Fig. 2.

Data for experimenting with the `driftR` package are available from within the software itself. Calling `sondeRaw` after loading the package and assigning it to an object in the global environment will provide users with a small sample data set. These data can be used to illustrate the utility of the remaining functions. Figure 3 provides a summary of a sample R session where each of the subsequent functions is implemented.

Creating correction factors

After the data are imported and formatted correctly, correction factors need to be generated using the `dr_factor()` function. This function is an implementation of Eq. 1. The result of executing `dr_factor()` is a new variable added to the data set that contains the appropriate correction factor for each time observation. The name of this new variable is defined by the user and will be used in the following steps of the data correction process.

Correcting the data

Once the incremental correction factors are generated, individual variables can be drift-corrected using either the `dr_correctOne()` or `dr_correctTwo()` functions, which represent one- or two-point calibration corrections, respectively (see Eq. 2 for one-point and Eqs. 3a–3c for two-point corrections). Executing either of these two functions will return a new variable, defined by the user, that contains the corrected data for the specified water quality parameter.

The `driftR` program deliberately creates a new variable during the correction process, rather than overwriting the uncorrected data, to ensure that each data set preserves a copy of the uncorrected data for quality control purposes and to maintain the ability to repeat the correction if incorrect values were initially used (e.g., the calibration standard for chloride was 1000 mg/L, but 100 mg/L was erroneously entered into a `driftR` correction function). This step should be repeated for each water quality parameter that needs to be corrected. It is important to note that the `dr_correctOne()` and `dr_correctTwo()`

Fig. 3 A sample driftR session

```

# load the driftR package
library(driftR)

# import sample data exported from a Sonde
waterTibble <- sondeRaw

# calculate correction factor and keep dateTime var
# results stored in new vector corrFac and dateTime
waterTibble <- dr_factor(waterTibble, corrFactor = corrFac, dateVar = Date,
                        timeVar = Time, keepDateTime = TRUE)

# apply one-point calibration to SpCond;
# results stored in new vector SpConde_Corr
waterTibble <- dr_correctOne(waterTibble, sourceVar =
                             SpCond, cleanVar = SpConde_Corr, calVal = 1.07,
                             calStd = 1, factorVar = corrFac)

# apply two-point calibration to pH;
# results stored in new vector pH_Corr
waterTibble <- dr_correctTwo(waterTibble, sourceVar = pH, cleanVar = pH_Corr,
                             calValLow = 7.01, calStdLow = 7,
                             calValHigh = 11.8, calStdHigh = 10,
                             factorVar = corrFac)

# drop observations to account for instrument equilibration
waterTibble <- dr_drop(waterTibble, head=10, tail=5)

#replace observations with NA for a date range
waterTibble <- dr_replace(waterTibble, sourceVar = pH, overwrite = TRUE,
                          dateVar = Date, timeVar = Time,
                          from = "2018-02-05", to = "2018-02-09")

```

functions can accept any water quality parameter. Thus, if a user employs a new water quality sensor, driftR will be able to correct the data it generates.

Dropping observations

Data observations often need to be removed from a data set because they are not representative of the in situ conditions for the aquatic system. For example, the instrument must be removed from and returned to the water for calibration. Consequently, the monitoring device may record several observations out of the water that will appear in the data set (see Fig. 4 for a visualization of this phenomenon in the uncorrected data as well as the corrected data with errant observations removed). Some sensors (e.g., ISE sensors) may also need time to equilibrate in the aquatic environment of interest after the instrument has been calibrated.

In both cases, these observations can be removed from the data set using the `dr_drop()` function (see Fig. 4). The function implements three distinct methods for dropping observations. The first method selectively removes observations from the beginning of a data set (i.e., the “head”) or the end of a data set (i.e., the “tail”), or both. The second method drops observations over a specified date range. Dates

in driftR can be specified in either month-day-year (“MDY”) or year-month-day (“YMD”) formatting, making driftR’s use in contexts outside of the USA seamless. The last method drops observations expressionally, meaning that if an observation meets a certain condition, then the observation will be dropped. Expressions are written using logical operators such as “>” (greater than), “<” (less than), “>=” (greater than or equal to), “<=” (less than or equal

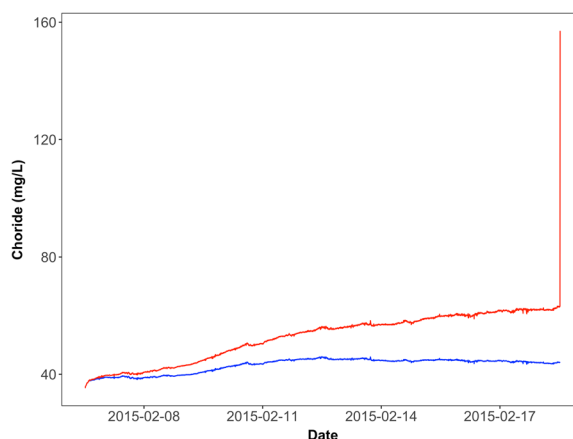
**Fig. 4** A plot of uncorrected (red) and corrected (blue) chloride data over a 12-day monitoring period. A total of 30 min of data were removed from the beginning and end of the data set using the `dr_drop()` function to account for the instrument being out of the water as it was dropped or removed for calibration

Fig. 5 A sample driftR session using the pipe operator (%>%)

```
# load the driftR package
library(driftR)

# import sample data exported from a Sonde
waterTibble <- sondeRaw

# calculate correction factors, apply corrections, drop observations,
# and replace observations
waterTibble <- waterTibble %>%
  dr_factor(corrFactor = corrFac, dateVar = Date, timeVar = Time,
            keepDateTime = TRUE) %>%
  dr_correctOne(sourceVar = SpCond, cleanVar = SpCond_Corr, calVal = 1.07,
                calStd = 1, factorVar = corrFac) %>%
  dr_correctTwo(sourceVar = pH, cleanVar = pH_Corr, calValLow = 7.01,
                calStdLow = 7, calValHigh = 11.8, calStdHigh = 10,
                factorVar = corrFac) %>%
  dr_drop(head=10, tail=5) %>%
  dr_replace(waterTibble, sourceVar = pH, overwrite = TRUE, dateVar = Date,
            timeVar = Time, from = "2018-02-05", to = "2018-02-09")
```

to), “==” (equal), and “!=” (not equal). Multiconditional expressions can also be written by utilizing “&” (and) as well as “|” (or) between expressional terms. For example, to drop observations where the pH is greater than 10 or the chloride is less than 0, we would write: `dr_drop(df, exp = pH > 10 | chloride < 0)`.

Replacing observations

In certain cases, individual sensors, rather than the entire instrument, may record inaccurate data during a monitoring period. For example, the pH sensor bulb may be damaged at some point during the deployment, but the other sensors are unaffected. In these instances, `dr_drop()` cannot be used because this function would drop all of the specified data, not solely the parameter of interest. The function `dr_replace()` removes the selected observations for a single parameter and replaces them with NA (not available), which R reads as “empty” or “missing.” There are two methods for replacing data. The first method is over a specified date range and the second method is via an expression as discussed in “[Dropping observations.](#)”

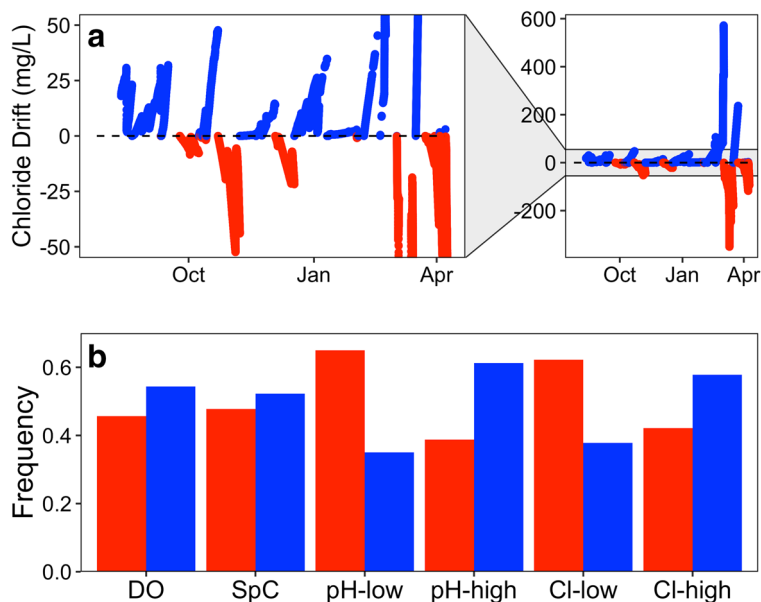
Piping functions together

The driftR functions `dr_factor()`, `dr_correctOne()`, `dr_correctTwo()`, `dr_drop()`, and `dr_replace()` utilize the argument `.data` to tell R in which data frame the data are stored. This argument allows functions to be combined with the pipe operator (%>%), which is a product of the magrittr package (Bache and Wickham 2014). The advantage of piped functions is that several scripts can be “piped” together in a way that makes them more readable, makes the data wrangling process more streamlined, and allows the functions to be used in conjunction with other packages via a common “tidy” data format (Wickham 2014). This behavior is common in the tidyverse ecosystem of R packages (Wickham 2017), for example. driftR’s implementation of the pipe operator means that all of the core functions can be chained together in a single pipeline (see Fig. 5). It also means that, for users already familiar with the tidyverse, driftR integrates seamlessly into pre-existing data wrangling and visualization workflows.

Table 3 WRTDS model results for corrected and uncorrected chloride data collected for an urban spring

Year	Flow-normalized concentration (mg/L)			Flow-normalized flux (× 10 ⁴ kg/year)		
	Uncorrected	Corrected	Difference (%)	Uncorrected	Corrected	Difference (%)
2015	339	257	32.0	11.4	9.3	22.6
2016	167	176	5.1	5.5	6.4	14.1

Fig. 6 **a** Chloride sensor drift over an 8-month period and **b** a histogram showing the frequency of positive and negative drifts for various water quality parameters, including DO, specific conductivity (SpC), pH, and chloride (Cl). Note that blue represents positive drift and red represents negative drift in both plots



The impact of drift on water quality models

Models that utilize water quality data are susceptible to large inaccuracies as a result of instrument drift and fouling. To illustrate this point, we used the USGS Weighted Regression on Time Discharge and Season (WRTDS) model (Hirsch et al. 2010) from the R package, EGRET, to determine the effects of drift on chloride flux calculations. We applied the model to 2 years of continuous corrected and uncorrected chloride data from an urban spring that experiences regular applications of winter de-icing salts (data are from Robinson and Hasenmueller (2017)). Using this model, we found that the flow-normalized chloride flux was 21,000 kg larger for the uncorrected compared with the corrected data in 2015 (22.6% difference) and 9,000 kg smaller for the uncorrected compared with the corrected data in 2016 (14.1% difference; Table 3). In this case, the chloride sensor experienced both positive (i.e., larger than true value) and negative (i.e., smaller than true value) drifts (see Fig. 6). That means that over short timescales of weeks to months, drift can significantly impact model results. Over long timescales of years to decades, drift errors in modeled results might be smaller than expected because positive drift can be offset by negative drift. Sensors can also consistently drift in one direction, leading to significant overestimations or underestimations of results on longer timescales. This highlights

the need to drift-correct data to increase model reliability. We also note that the frequency of positive and negative drifts differed between the low and high standards used in two-point calibrations (Fig. 6b), demonstrating the importance of employing multiple calibration standards for certain parameters.

Using `driftR` with other R packages

An advantage of implementing continuous water quality monitoring data correction processes in R is that data can be corrected and analyzed in a single session. Though `driftR` does not offer tools for analysis, it can be used in conjunction with other R packages to increase functionality. After drift-correcting data, the corrected data can be visualized using the package `ggplot2` (Wickham 2009). The `ggplot2` package offers a wide range of useful visualization capabilities including the ability to create faceted or stacked plots of various water quality parameters. In addition to data visualization, statistical analyses can be executed in R. The `stats` package offers functions for *t* tests, ANOVAs, and other descriptive statistics. Water quality data modeling can also be implemented in R by numerous packages, including `caret` for predictive modeling (Kuhn 2018), `neuralnet` for neural network models (Fritsch and Guenther 2016), `randomForest` for random forest models (Liau

and Wiener 2002), and `xgboost` for gradient boosted trees (Chen et al. 2018). Indeed, `driftR` complements a suite of packages in R by filling the need for pre-analysis drift correction, which is an important step in ensuring accurate analyses thereafter. By combining packages, all of the analyses for water quality research projects can be implemented and documented in R.

Conclusion

Continuous water quality monitoring instruments are used in a wide variety of studies and offer high-resolution chemical and physical data sets for aquatic systems over time. The sensors on these instruments are susceptible to errors due to drift and/or fouling after calibration. This drift can significantly alter a data set, with some types of sensors experiencing > 200% drift over monitoring periods of a few weeks. There are several programs for water quality data drift correction, but these programs are expensive, not publicly available, and/or lack user control. There is a need in the scientific community for a free and widely available software package for drift-correcting water quality data sets. The `driftR` package is a novel R program that uses functions that interpolate a linear drift correction over time and includes both one- and two-point variable data corrections. `driftR` is free, offers the user control over the drift correction process, can be used with any continuous water quality monitoring instrument, accepts any water quality parameter that needs to be corrected, and uses methods that are reproducible by other scientists.

Acknowledgments We thank numerous field assistants for maintaining the continuous water quality monitoring instruments that obtained the data used to optimize the `driftR` package: Michael Abegg, Mary Arenberg, Camille Buckley, Emily Deebea, Armahni Fearn, Margaret Hennessey, Kayla Lockmiller, Theresa Martin, David Pan, and Heather Robinson. We also thank the developers of the software that `driftR` is built upon, particularly Hadley Wickham, for their contributions to the R community.

References

- Appling, A.P., Leon, M.C., McDowell, W.H. (2015). Reducing bias and quantifying uncertainty in watershed flux estimates: the `r` package `loadflex`. *Ecosphere*, 6(12), 1–25.
- Bache, S.M., & Wickham, H. (2014). `Magrittr`: A Forward-Pipe Operator for R. R package version 1.5.
- Beck, M.W. (2016). `Swmpr`: an `r` package for retrieving, organizing, and analyzing environmental data for estuaries. *R Journal*, 8(1), 219.
- Bhaskar, A.S., & Welty, C. (2015). Analysis of subsurface storage and streamflow generation in urban watersheds. *Water Resources Research*, 51(3), 1493–1513.
- Buck, S. (2015). Solving reproducibility.
- Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y. (2018). `xgboost`: Extreme Gradient Boosting. R package version 0.71.2.
- Delauney, L., Compere, C., Lehaitre, M. (2010). Biofouling protection for marine environmental sensors. *Ocean Science*, 6(2), 503–511.
- Fritsch, S., & Guenther, F. (2016). `Neuralnet`: Training of Neural Networks. R package version 1.33.
- Glibert, P.M., Kelly, V., Alexander, J., Codispoti, L.A., Boicourt, W.C., Trice, T.M., Michael, B. (2008). In situ nutrient monitoring: a tool for capturing nutrient variability and the antecedent conditions that support algal blooms. *Harmful Algae*, 8(1), 175–181.
- Hasenmueller, E.A. (2011). The hydrology and geochemistry of urban and rural watersheds in east-central Missouri. Washington University in St. Louis.
- Hasenmueller, E.A., Criss, R.E., Winston, W.E., Shaughnessy, A.R. (2017). Stream hydrology and geochemistry along a rural to urban land use gradient. *Applied Geochemistry*, 83, 136–149.
- Hirsch, R.M., & De Cicco, L.A. (2015). User guide to exploration and graphics for river trends (`egret`) and data retrieval: R packages for hydrologic data. Tech. rep., US Geological Survey.
- Hirsch, R.M., Moyer, D.L., Archfield, S.A. (2010). Weighted regressions on time, discharge, and season (`wrtds`), with an application to chesapeake bay river inputs I. *JAWRA Journal of the American Water Resources Association*, 46(5), 857–880.
- Horsburgh, J.S., Reeder, S.L., Jones, A.S., Meline, J. (2015). Open source software for visualization and quality control of continuous hydrologic and water quality sensor data. *Environmental Modelling & Software*, 70, 32–44.
- Jimenez-Montealegre, R., Verdegem, M., Van Dam, A., Verreth, J. (2002). Conceptualization and validation of a dynamic model for the simulation of nitrogen transformations and fluxes in fish ponds. *Ecological modelling*, 147(2), 123–152.
- Kuhn, M. (2018). `Caret`: Classification and Regression Training. R package version 6.0-80.
- Liaw, A., & Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3), 18–22.
- Lowndes, J.S.S., Best, B.D., Scarborough, C., Afflerbach, J.C., Frazier, M.R., O'Hara, C.C., Jiang, N., Halpern, B.S. (2017). Our path to better science in less time using open data science tools. *Nature ecology & evolution*, 1(6), 0160.
- Pellerin, B.A., Wollheim, W.M., Feng, X., Vörösmarty, C.J. (2008). The application of electrical conductivity as a tracer for hydrograph separation in urban catchments. *Hydrological Processes*, 22(12), 1810–1818.

- Roberts, B.J., Mulholland, P.J., Hill, W.R. (2007). Multiple scales of temporal variability in ecosystem metabolism rates: results from 2 years of continuous monitoring in a forested headwater stream. *Ecosystems*, 10(4), 588–606.
- Robinson, H.K., & Hasenmueller, E.A. (2017). Transport of road salt contamination in karst aquifers and soils over multiple timescales. *Science of The Total Environment*, 603, 94–108.
- Ryberg, K.R., & Vecchia, A.V. (2012). Waterdata—an r package for retrieval, analysis, and anomaly calculation of daily hydrologic time series data, version 1.0. Tech. rep., US Geological Survey.
- Wagner, R.J., Boulger, R.W. Jr., Oblinger, C.J., Smith, B.A. (2006). Guidelines and standard procedures for continuous water-quality monitors: station operation, record computation, and data reporting. Tech. rep.
- Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York: Springer.
- Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59(10), 1–23.
- Wickham, H. (2017). Tidyverse: Easily Install and Load the 'Tidyverse'. R package version 1.2.1.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.